

ACCELERATION OF QUASI-MONTE CARLO
APPROXIMATION

Joseph S. Severino, Jr.

Texas Tech University

ACKNOWLEDGEMENTS

First of all, I would like to thank Dr. Allen. He has been a tremendous help through his tireless efforts from the start of this thesis to its finish. Next I thank my parents for their continued support throughout my academic career. I certainly could not have made it this far without them. I would also like to thank Dr. Victory, for sitting on my thesis committee as well as, with Dr. Allen, providing financial support for my research. Dr. Seshaiyer deserves credit for helping with my presentations, through which I gained a better understanding of what I am doing. Additionally, I thank my fellow graduate students, particularly David Martin, Daniel Smith, Maggie Barclay, Cindy Martin, Cody Farmer, Keith Nabb, Jesse Fagan, Amanda Hummer, Jeff Garza, Allen Miller, Angela Menke, Church Mickel, and Doug Meador. They helped keep a relaxed atmosphere around our offices and continually distracted me from doing work, for which I am thankful.

Also deserving of thanks are my friends with whom I spent my weekends, including Heath Garner, Julian Hooker, Gye Kraemer, Christina Caudle Kraemer, Bill Ferguson, and Laurel Guernsey. Their company was invaluable, and without a doubt they helped ease the stress of otherwise difficult semesters.

Lastly, I would like to acknowledge some of the musicians whose works accompanied me throughout the thesis process: Richard D. James, John Digweed, Ken Jordan, Scott Kirkland, Rupert Parkes, Bill Leeb, Rhys Fulber, Chris Petersen, Damian Higgins, Edgar Froese, and Paul Haslinger.

CONTENTS

ACKNOWLEDGEMENTS	ii
ABSTRACT	iv
LIST OF FIGURES	v
LIST OF TABLES	vi
I INTRODUCTION	1
II MOTIVATION FOR QUASI-MONTE CARLO	3
III QUASI-MONTE CARLO SEQUENCES	5
IV DERIVATION OF ACCELERATION METHOD	12
V ERROR ANALYSIS	14
VI RESULTS	17
6.1 First Integral Problem	17
6.2 Second Integral Problem	23
6.3 Third integral Problem	31
VII CONCLUSION	36
BIBLIOGRAPHY	37
APPENDIX A. PROGRAM QMFAURE1.F	38
APPENDIX B. PROGRAM QMHALTON2.F	41
APPENDIX C. PROGRAM QMPROB3.F	43

ABSTRACT

In this paper, a new method is presented for numerically approximating integrals using quasi-Monte Carlo sequences. By applying a least-squares smoothing procedure to the sequences, a faster rate of convergence is achieved thus reducing the number of nodes required for the same degree of accuracy.

This acceleration method is applied to three particular integrals in a variety of dimensions. The first integral is a smooth exponential function and the second has a discontinuous integrand. The last integral deals with a specific problems in mathematical finance, computing the price for a European call option.

LIST OF FIGURES

3.1	256 Faure points in 2 dimensions.	8
3.2	1024 Faure points in 2 dimensions.	8
3.3	4096 Faure points in 2 dimensions.	9
3.4	16384 Faure points in 2 dimensions.	9
3.5	256 Halton points in 2 dimensions.	10
3.6	1024 Halton points in 2 dimensions.	10
3.7	4096 Halton points in 2 dimensions.	11
3.8	16384 Halton points in 2 dimensions.	11
6.1	First Integral for $s = 1$	17
6.2	First Integral for the Halton Sequence with $s = 2$	20
6.3	First Integral for the Faure Sequence with $s = 2$	20
6.4	First Integral for the Halton Sequence with $s = 4$	21
6.5	First Integral for the Faure Sequence with $s = 4$	21
6.6	First Integral for the Halton Sequence with $s = 6$	22
6.7	First Integral for the Faure Sequence with $s = 6$	22
6.8	First Integral for $s = 1$	24
6.9	Second Integral for the Halton Sequence with $s = 2$	28
6.10	Second Integral for the Faure Sequence with $s = 2$	28
6.11	Second Integral for the Halton Sequence with $s = 4$	29
6.12	Second Integral for the Faure Sequence with $s = 4$	29
6.13	Second Integral for the Halton Sequence with $s = 6$	30
6.14	Second Integral for the Faure Sequence with $s = 6$	30
6.15	Third Integral Results	35

LIST OF TABLES

6.1	Results For First Integral in Two Dimensions	18
6.2	Results For First Integral in Two Dimensions, Continued	19
6.3	Results For Second Integral in Two Dimensions	24
6.4	Results For Second Integral in Two Dimensions, Continued	25
6.5	Results For Second Integral in Two Dimensions, Continued	26
6.6	Results For Third Integral	32
6.7	Results For Third Integral, Continued	33
6.8	Results For Third Integral, Continued	34

CHAPTER I
INTRODUCTION

In this document a method for accelerating quasi-Monte Carlo approximations is presented. Through the application of a least-squares smoothing procedure a faster rate of convergence is achieved, reducing the number of nodes required for a prescribed degree of accuracy. This method of acceleration will be described and its approximation error will be analyzed as well. Four specific integrals will be examined using two different quasi-Monte Carlo sequences, the Faure sequence and the Halton sequence. For each integral a variety of dimensions will be considered, and the results for each sequence will be compared with the corresponding accelerated sequence.

The motivation for this paper is to find a more efficient method for the numerical approximation of integrals, with special consideration given to mathematical finance problems.

The first integral to be considered is a smooth exponential equation,

$$\int_0^1 \int_0^1 \cdots \int_0^1 \exp(x_1 x_2 \cdots x_s) dx_1 dx_2 \cdots dx_s, \text{ for } s = 1, 2, 4, 6 \quad (1.1)$$

where s is the dimension of the integral.

The second integral, studied in [3], has a discontinuous integrand.

$$\int_0^1 \int_0^1 \cdots \int_0^1 \left(\frac{1}{R_s}\right)^s \chi_s(x_1, x_2, \dots, x_s) dx_1 dx_2 \cdots dx_s \quad (1.2)$$

where

$$R_s = \left(\frac{7}{8}\right)^{\frac{6}{s}}$$

and

$$\chi_s(x_1, x_2, \dots, x_s) = \begin{cases} 1 & \text{if } |x_i - .5| \leq \frac{R_s}{2}, \text{ for } i = 1, 2, \dots \\ 0 & \text{otherwise.} \end{cases}$$

and s is the dimension.

The third integral, studied in [2], determines the price of a European call option and is strictly a one-dimensional problem.

$$C = \frac{e^{-rT}}{\sqrt{2\pi\sigma^2T}} \int_{\ln s}^{\infty} (e^x - S) e^{\frac{-(x - \ln S - rT + \frac{\sigma^2T}{2})}{2\sigma^2T}} dx \quad (1.3)$$

where

$$T = 1, r = 0.1, S = 100, \sigma = 0.3, \text{ and } s = 1$$

CHAPTER II
MOTIVATION FOR QUASI-MONTE CARLO

In this chapter, the advantages of using quasi-Monte Carlo over numerical integration and standard Monte Carlo methods will be discussed.

For numerical integration, consider, for example, the trapezoid rule in one dimension. The trapezoid rule yields the approximation

$$\int_0^1 f(u) du \approx \sum_{n=0}^m w_n f\left(\frac{n}{m}\right) \quad (2.1)$$

where the weights are $w_0 = w_m = \frac{1}{2m}$ and $w_n = \frac{1}{m}$ for $1 \leq n \leq m - 1$. By assuming now that $f \in C^2[0, 1]$, the error is then $O(m^{-2})$, as shown in [4]. Considering general dimension s , the trapezoid rule has the approximation

$$\int_0^1 \cdots \int_0^1 f(\vec{u}) d\vec{u} \approx \sum_{n_1=0}^m \cdots \sum_{n_s=0}^m w_{n_1} \cdots w_{n_s} f\left(\frac{n_1}{m}, \dots, \frac{n_s}{m}\right) \quad (2.2)$$

Here, the total number of nodes used is $N = (m + 1)^s$, and the error is once again $O(m^{-2})$. In terms of N , this becomes $O(N^{-\frac{2}{s}})$, which means that in order to have an error of less than 10^{-4} in a six dimensional problem, approximately, 10^{12} nodes would be required. A similar analysis applies to any standard numerical integration method.

The standard Monte Carlo method will be examined next. Consider the integral

$$\int_B f(\vec{u}) d\vec{u} = \left(\int_B d\vec{u} \right) E(f) \quad (2.3)$$

where

$$E(f) = \frac{\int_B f(\vec{u}) d\vec{u}}{\int_B d\vec{u}} \quad (2.4)$$

is the expected value of f , B is a subset of I^s , and $I = [0, 1]$. The Monte Carlo estimate for $E(f)$ is found by taking N independent random samples and letting

$$E(f) \approx \frac{1}{N} \sum_{n=1}^N f(a_n) \quad (2.5)$$

The Central Limit Theorem then guarantees that

$$\lim_{n \rightarrow \infty} \frac{1}{N} \sum_{n=1}^N f(a_n) = E(f) \quad (2.6)$$

Next, the variance of f is

$$\sigma^2(f) = \int_B (f - E(f))^2 d\vec{u} \quad (2.7)$$

Once again by the Central Limit Theorem, for large N , it is obtained that

$$\text{prob} \left(\left| \frac{1}{N} \sum_{n=1}^N f(a_n) - \int_B f(\vec{u}) d\vec{u} \right| \leq \frac{\lambda\sigma}{\sqrt{N}} \right) \approx \frac{1}{\sqrt{2\pi}} \int_{-\lambda}^{\lambda} e^{-\frac{x^2}{2}} dx \quad (2.8)$$

So for example, if $\lambda = 1.96$, then the error is less than $\frac{1.96\sigma}{\sqrt{N}}$ with a probability of 0.95. Thus, it is said that the standard Monte Carlo method has error proportional to $\frac{1}{\sqrt{N}}$. Here, the error does not depend on the dimension of the integral being approximated. Thus if 10^{12} nodes were used as before, the error for Monte Carlo would be approximately 10^{-6} , which is better than the trapezoid rule by a factor of 10^{-2} .

Considering quasi-Monte Carlo now, let

$$\int_B f(\vec{u}) d\vec{u} \approx \frac{1}{N} \sum_{n=1}^N f(\vec{x}_n) \quad (2.9)$$

where $\vec{x}_n \in B$ are deterministic points. By constructing sets of deterministic nodes, an error bound of $O(N^{-1}(\log N)^{s-1})$ is possible. This error bound and its derivation will be discussed in depth in Chapter V. It will later be shown in the results in Chapter VI that $O(N^{-1}(\log N))$ is a reasonable estimate for quasi-Monte Carlo approximation. The error is actually of the form

$$N^{-1}[c_1(\log N) + c_2(\log N)^2 + \dots + c_{s-1}(\log N)^{s-1}], \quad (2.10)$$

but it turns out that the coefficient c_1 is typically dominant, whereas the remaining coefficients are small. Thus if it is assumed that the error is $O(N^{-1}(\log N))$, then if $N = 10^{12}$ as before, the error is approximately 10^{-11} . So it can be concluded that in general, quasi-Monte Carlo approximation is more accurate than standard Monte Carlo and numerical integration, particularly in high dimensions.

CHAPTER III
QUASI-MONTE CARLO SEQUENCES

In this chapter, three types of quasi-Monte Carlo sequences will be described: van der Corput, Faure, and Halton. Van der Corput sequences are specifically one dimensional and form the basis for the Faure and Halton sequences. The primary focus of this chapter is how the Faure and Halton sequences are formed.

To begin, the generation of a one-dimensional sequence of quasi-random numbers is delineated. Let r be any prime number. Any base 10 integer n has a unique expansion in base r , and this expansion has only a finite number of nonzero terms. The number of terms in the expansion will be denoted m , and in general,

$$m = g(\log_r n) \tag{3.1}$$

where g represents the greatest integer function. For programming purposes, it is not necessary that this value be precisely known, though it may help for memory allocation. The quasi-random number corresponding to n will be in the set $[0, 1]$. It is formed by reflecting the base r expansion about the decimal point and then converting that result back to base 10.

For example, let $r = 2$ and $n = 6$. The first step is to convert 6 to a base 2 number.

$$6 = 1(2^2) + 1(2^1) + 0(2^0) \rightarrow 110_2 \tag{3.2}$$

Next, the base 2 number is reflected about the binary point.

$$110_2 \rightarrow .011_2 \tag{3.3}$$

Last, this reflected number is converted back into base 10.

$$\begin{aligned} .011_2 &\rightarrow 0(2^{-1}) + 1(2^{-2}) + 1(2^{-3}) \\ &= 0 + \frac{1}{4} + \frac{1}{8} \\ &= \frac{3}{8} \end{aligned} \tag{3.4}$$

The first ten numbers of the one dimensional sequence with $r = 2$ are

$$\frac{1}{2}, \frac{1}{4}, \frac{3}{4}, \frac{1}{8}, \frac{5}{8}, \frac{3}{8}, \frac{7}{8}, \frac{1}{16}, \frac{9}{16}, \frac{5}{16} \quad (3.5)$$

Thus, a pattern can be discerned from the sequence. Not including the endpoints, the sequence begins with $\frac{1}{2}$, then fills in the remaining two fourths, the remaining four eighths, the remaining eight sixteenths, and so on. A similar pattern is observed for other values of r . So it can be seen that the sequence is slowly and evenly filling in the space $[0, 1]$. Such sequences are called van der Corput sequences, and can be formed using any values of r and n .

Now it is easy to describe the formation of a one dimensional Faure sequence. The value of r for a Faure sequence is always taken to be the smallest prime number greater than or equal to the dimension of the desired quasi-random number, and in a one dimensional sequence this will always be 2. Therefore the list of numbers in equation 3.5 represents the first ten one dimensional Faure numbers because they were generated with $r = 2$. In general, for one dimension,

$$n = \sum_{j=0}^m a_j(n)r^j \quad (3.6)$$

where the $a_j(n)$ are the coefficients of the base r expansion of n , with $j = 0$ corresponding to the ones position. Then the n th quasi-random number in the sequence is

$$\Phi_r(n) = \sum_{j=0}^m a_j(n)r^{-j-1} \quad (3.7)$$

Multi-dimensional Faure sequences can be generated recursively using the one dimensional sequence with an appropriate value of r . For some value of n , first the coefficients of the base r expansion are found, i.e. the values of $a_j^1(n)$ for $j = 0 \dots m$. The superscript indicates the dimension. Now assuming the values of $a_j^{k-1}(n)$ are known, the terms of the next higher dimension are generated by [2]:

$$a_j^k(n) = \sum_{j=0}^m \sum_{i=j}^m [({}^i C_j \cdot a_i^{k-1}(n)) \bmod r] \quad (3.8)$$

where

$${}^iC_j = \frac{i!}{j!(i-j)!} \quad (3.9)$$

Then the corresponding points in the Faure sequence are found via

$$\Phi_r^k(n) = \sum_{j=0}^m a_j^k(n) r^{-j-1} \quad (3.10)$$

Halton sequences require a simpler algorithm than Faure sequences. A one dimensional Halton sequence is the van der Corput sequence formed using $r = 2$. It turns out that the one dimensional Halton sequence is exactly the same as the one dimensional Faure sequence.

Now consider s -dimensional Halton points. Let p_1, \dots, p_s represent the first s prime numbers and let $x(n) = (x_1, \dots, x_s)$ be the n th s -dimensional Halton point. Then x_i is the n th van der Corput point with $r = p_i$, for $i = 1, \dots, s$. Thus, for a three dimensional Halton point, the x-coordinates are van der Corput points with $r = 2$, the y-coordinates are van der Corput points with $r = 3$, and the z-coordinates are van der Corput points with $r = 5$.

The following four pages contain eight figures that demonstrate how the Faure and Halton sequences fill in the unit square. In the case of the Faure sequence, a certain amount of regularity is noticed in the position of the points, due to the fact that $r = 2$ would be used for both one and two dimensions. For Halton, however, the x and y coordinates are found using two different values of r , and so the pattern is not as pronounced. These graphs were created by outputting data from a FORTRAN program into a MATLAB program file that was then executed in MATLAB.

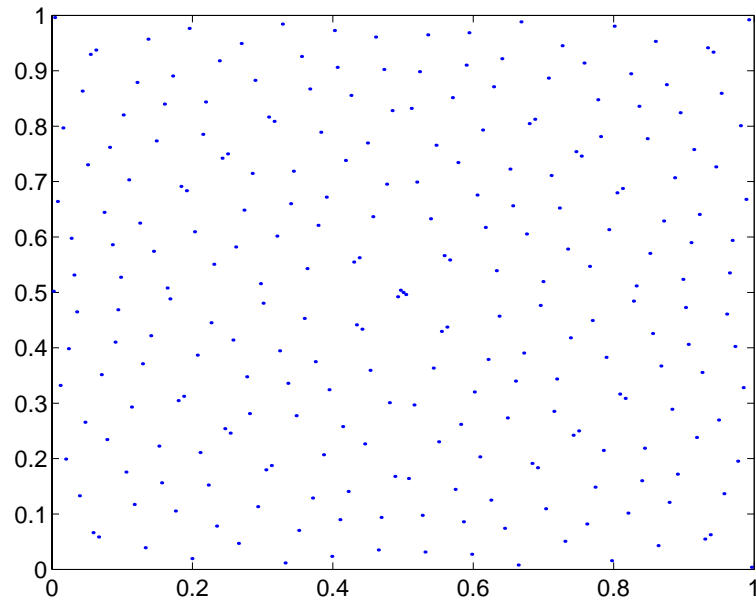


Figure 3.1: 256 Faure points in 2 dimensions.

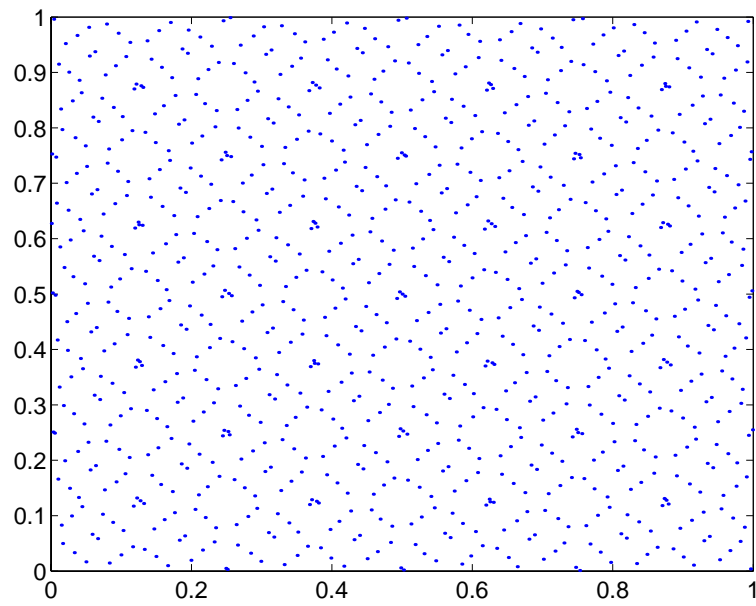


Figure 3.2: 1024 Faure points in 2 dimensions.

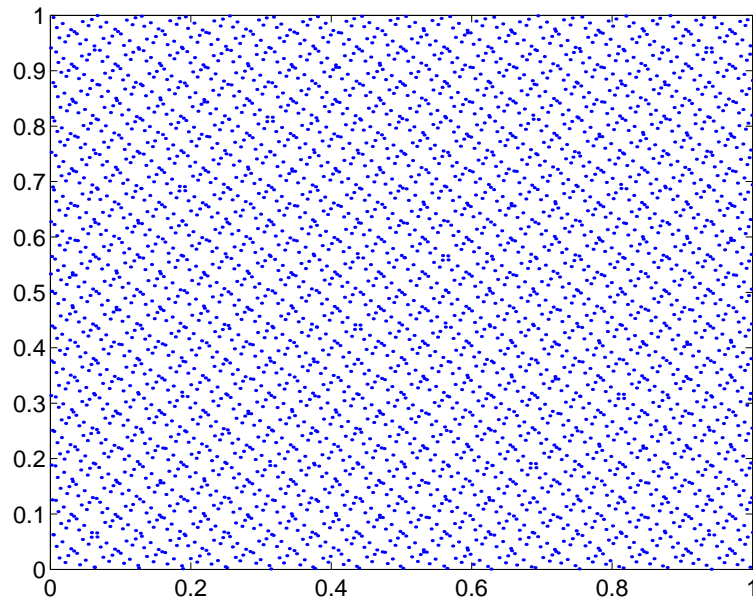


Figure 3.3: 4096 Faure points in 2 dimensions.

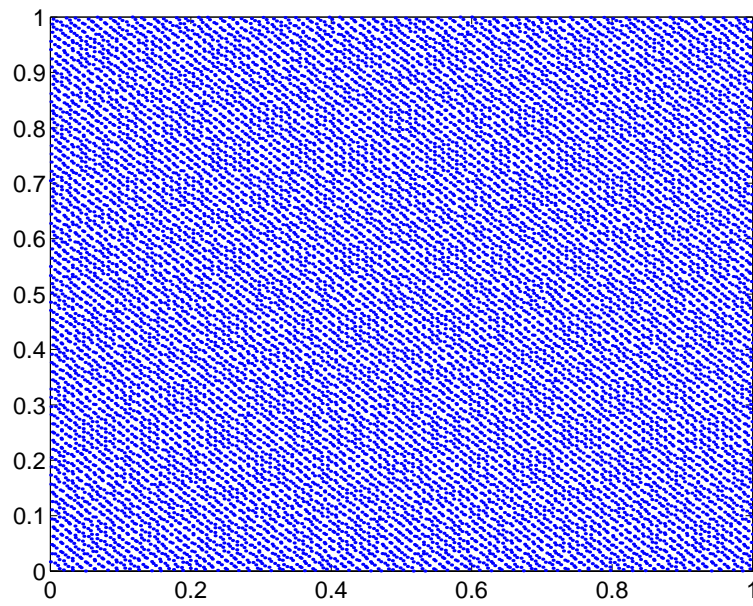


Figure 3.4: 16384 Faure points in 2 dimensions.

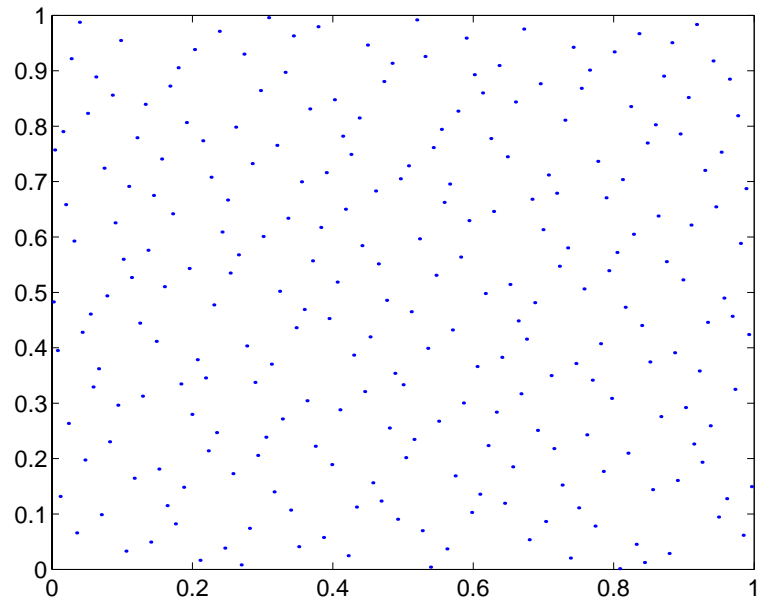


Figure 3.5: 256 Halton points in 2 dimensions.

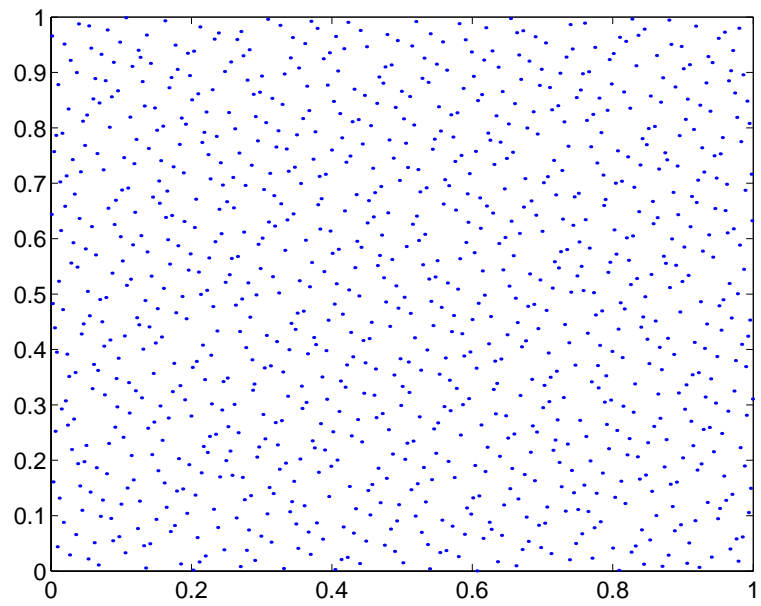


Figure 3.6: 1024 Halton points in 2 dimensions.

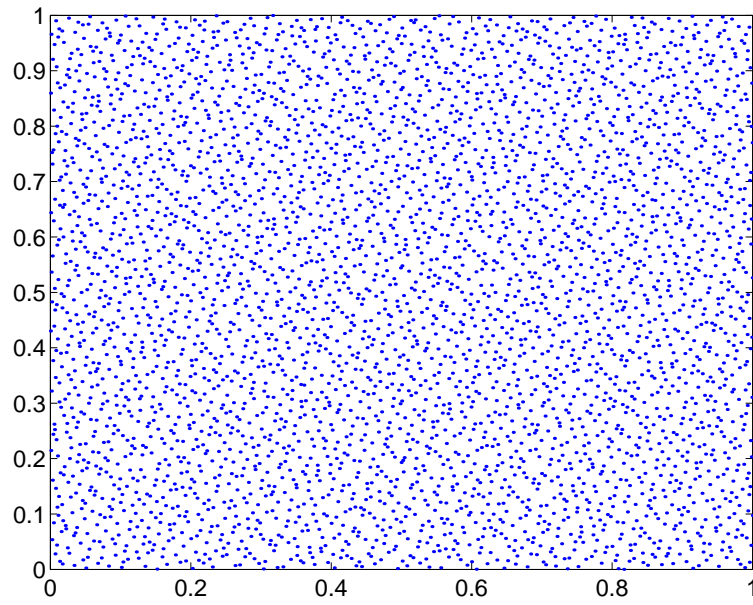


Figure 3.7: 4096 Halton points in 2 dimensions.

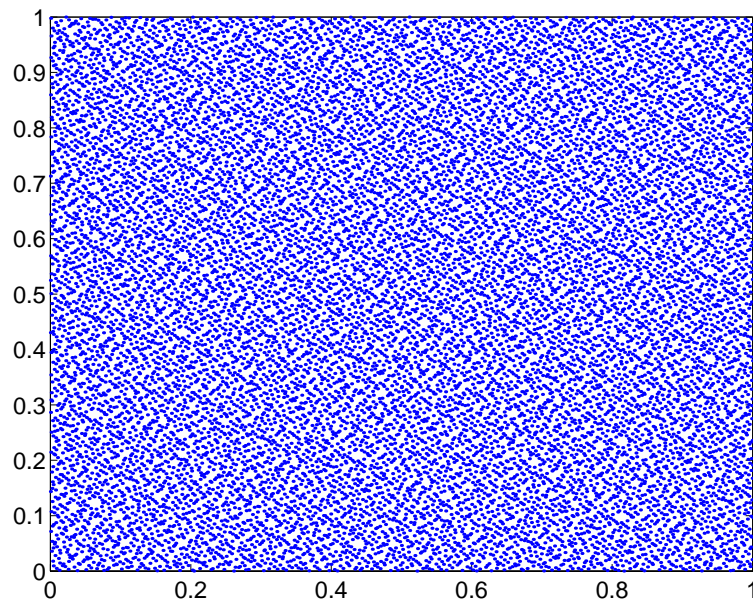


Figure 3.8: 16384 Halton points in 2 dimensions.

CHAPTER IV
DERIVATION OF ACCELERATION METHOD

This chapter describes how the acceleration method used was obtained. Let

$$a = \int_{I^s} f(\vec{u})d\vec{u} \approx \frac{1}{n} \sum_{m=1}^n f(\vec{u}_m) = w_n \quad (4.1)$$

where w_n is the n th approximation of a using some quasi-Monte Carlo sequence $\{\vec{u}_m\}_{m=1}^n$. By the theory of low discrepancy sequences, as described in (insert citation here), it is reasonable to assume an error expansion as follows.

$$w_n = a + b_{0,n} \frac{\log n}{n} + b_{1,n} \frac{(\log n)^2}{n} + \dots + b_{s,n} \frac{(\log n)^{s+1}}{n}, \text{ for } n \geq 2 \quad (4.2)$$

Now assume that the $\frac{\log n}{n}$ term is large. This is a good assumption for s small or function f smooth. The goal now is to estimate

$$a = \int_{I^s} f(\vec{u})d\vec{u} \quad (4.3)$$

Let

$$w_n = a + (b + \varepsilon_n) \frac{\log n}{n} + \gamma_n \frac{(\log n)^\alpha}{n} \quad (4.4)$$

where $|\varepsilon_n| < \varepsilon$ and $|\gamma_n| < \gamma$ for $2 \leq \alpha \leq s$. Now, assuming that γ is small,

$$w_n \approx a + (b + \varepsilon_n) \frac{\log n}{n}, \text{ for } n \geq 2 \quad (4.5)$$

Now it is necessary to put 4.5 into the form for a standard least squares procedure. Multiplying both sides by $\frac{n}{\log n}$, the result is

$$\frac{nw_n}{\log n} = a \frac{n}{\log n} + b + \varepsilon_n \quad (4.6)$$

Letting $y_n = \frac{nw_n}{\log n}$ and $x_n = \frac{n}{\log n}$,

$$y_n = ax_n + b + \varepsilon_n \quad (4.7)$$

Then, by adding and subtracting $a\bar{x}$, the equation becomes

$$y_n = a(x_n - \bar{x}) + (a\bar{x} + b) + \varepsilon_n \quad (4.8)$$

where

$$\bar{x} = \frac{\sum_{n=2}^N x_n}{N-1}, \quad (4.9)$$

assuming there are N points. Next, each side of equation 4.8 is multiplied by $(x_n - \bar{x})$ and then it is solved for a . Since ε is small, the terms that include it can be discarded. Notice that $(a\bar{x} + b)$ is just a constant. Also,

$$\begin{aligned} \sum_{n=2}^N (x_n - \bar{x}) &= \sum_{n=2}^N x_n - \sum_{n=2}^N \bar{x} \\ &= \sum_{n=2}^N x_n - \bar{x} \sum_{n=2}^N 1 \\ &= \sum_{n=2}^N x_n - (N-1)\bar{x} \\ &= \sum_{n=2}^N x_n - (N-1) \left(\frac{1}{N-1} \sum_{n=2}^N x_n \right) \\ &= 0 \end{aligned} \quad (4.10)$$

Then

$$\sum_{n=2}^N (a\bar{x} + b)(x_n - \bar{x}) = 0 \quad (4.11)$$

Thus,

$$a \approx \frac{\sum_{n=2}^N y_n (x_n - \bar{x})}{\sum_{n=2}^N (x_n - \bar{x})^2}, \quad (4.12)$$

where $x_n = \frac{n}{\log n}$, $y_n = w_n x_n$, and w_n is the n th quasi-Monte Carlo approximation of a . This approximation can in turn be used to program the accelerated method used in this thesis, as shown in the equations included in the appendices.

CHAPTER V
ERROR ANALYSIS

Now it is necessary to examine the error in the acceleration method described. Referring back to equation 4.8, now include the γ_n term that was previously discarded. Letting $c = b + a\bar{x}$ and $\beta_n = \gamma_n(\log n)^{\alpha-1}$, the equation becomes

$$y_n = a(x_n - \bar{x}) + c + \varepsilon_n + \beta_n \quad (5.1)$$

Note that β_n represents the higher order terms which were postulated as being small. Then

$$a \approx \hat{a} = \frac{\sum_{n=2}^N y_n(x_n - \bar{x})}{\sum_{n=2}^N (x_n - \bar{x})^2} \approx \frac{\sum_{n=2}^N \beta_n(x_n - \bar{x})}{\sum_{n=2}^N (x_n - \bar{x})^2} + a \quad (5.2)$$

and, by applying the Cauchy-Schwarz inequality,

$$|E_N| = \left| \frac{\sum_{n=2}^N \beta_n(x_n - \bar{x})}{\sum_{n=2}^N (x_n - \bar{x})^2} \right| \leq \frac{\left(\sum_{n=2}^N \beta_n^2 \right)^{\frac{1}{2}}}{\left(\sum_{n=2}^N (x_n - \bar{x})^2 \right)^{\frac{1}{2}}} \quad (5.3)$$

where E_N is the error.

Now it is necessary to show that

$$|E_n| \leq \frac{\hat{c}(\log N)^\alpha}{N} \quad (5.4)$$

This can be done by examining the numerator and denominator of equation 5.3 separately.

First, consider the numerator of 5.3.

$$\begin{aligned}
\left(\sum_{n=2}^N \beta_n^2\right)^{\frac{1}{2}} &\leq \gamma \left(\sum_{n=2}^N (\log n)^{2\alpha-2}\right)^{\frac{1}{2}} \\
&\leq \gamma \left(\int_2^{N+1} (\log x)^{2\alpha-2} dx\right)^{\frac{1}{2}} \\
&\leq \gamma \left(x(\log x)^{2\alpha-2}\Big|_2^{N+1}\right)^{\frac{1}{2}} \\
&\leq \gamma(N+1)^{\frac{1}{2}}(\log(N+1))^{\alpha-1}
\end{aligned} \tag{5.5}$$

Second, consider the denominator of 5.3,

$$\sum_{n=2}^N (x_n - \bar{x})^2 = \sum_{n=2}^N x_n^2 - (N-1)(\bar{x})^2 \tag{5.6}$$

However,

$$\begin{aligned}
\sum_{n=2}^N x_n^2 &\geq \int_2^N \frac{x^2}{(\log x)^2} dx \\
&\geq \frac{N^3}{3(\log N)^2} - \frac{2^3}{3(\log N)^2} \\
&\geq \frac{N^3}{3(\log(N+1))^2} - \frac{8}{3(\log(N+1))^2}
\end{aligned} \tag{5.7}$$

Also,

$$\bar{x} = \frac{1}{N-1} \sum_{n=2}^N \frac{n}{\log n} \leq \frac{1}{N-1} \int_2^{N+1} \frac{x}{\log x} dx \tag{5.8}$$

But

$$\int_2^{N+1} \frac{x}{\log x} dx \leq \frac{(N+1)^2}{b \log(N+1)} \tag{5.9}$$

for some $b < 2$ and N sufficiently large. Thus,

$$\bar{x} \leq \frac{1}{N-1} \cdot \frac{(N+1)^2}{b \log(N+1)} \tag{5.10}$$

To see this, first let $M = N + 1$. Suppose that

$$h(M) = \frac{M^2}{b \log M} - \int_2^M \frac{x}{\log x} dx \tag{5.11}$$

and

$$h'(M) = \left(\frac{2}{b} - 1\right) \frac{M}{\log M} - \frac{M}{b (\log M)^2} \quad (5.12)$$

Notice that at $M = \exp\left(\frac{1}{2-b}\right)$, $h'(M) = 0$. Also, $h'(M) > 0$ for $M > \exp\left(\frac{1}{2-b}\right)$ and $\lim_{M \rightarrow \infty} h'(M) = +\infty$. This implies that $h(M) \geq 0$ for M sufficiently large. For example, if $b = 1.75$, then $h'(M) > 0$ for $M > 55$. It can then be shown that $h(M) > 0$ for $M > 110$.

Hence,

$$\begin{aligned} \sum_{n=2}^N (x_n - \bar{x})^2 &\geq \frac{N^3}{3(\log(N+1))^2} - \frac{8}{3(\log(N+1))^2} - \frac{1}{N-1} \cdot \frac{(N+1)^4}{b^2 (\log(N+1))^2} \\ &= \frac{1}{(\log(N+1))^2} \left(\frac{N^3}{3} - \frac{8}{3} - \frac{(N+1)^4}{(N-1)b^2} \right) \\ &= \frac{N^3}{(\log(N+1))^2} \left(\frac{1}{3} - \frac{8}{3N^3} - \frac{(N+1)^4}{N^3(N-1)b^2} \right) \\ &\geq \frac{N^3}{(\log(N+1))^2} \left(\frac{1}{3} - \frac{1.013}{b^2} \right) \end{aligned} \quad (5.13)$$

for $N > 500$ and $1.75 \leq b < 2$

Thus

$$\sum_{n=2}^N (x_n - \bar{x})^2 \geq \frac{N^3}{(\log(N+1))^2} \left(\frac{1}{3} - \frac{1.013}{b^2} \right) \quad (5.14)$$

for N sufficiently large and $1.75 \leq b < 2$.

Therefore,

$$|E_N| \leq \frac{\gamma(N+1)^{\frac{1}{2}} (\log(N+1))^\alpha}{N^{\frac{3}{2}} \left(\frac{1}{3} - \frac{1.013}{b^2}\right)^{\frac{1}{2}}} \quad (5.15)$$

for N sufficiently large and $1.75 \leq b < 2$. Then,

$$|E_N| \leq \frac{(1.1)\gamma}{\left(\frac{1}{3} - \frac{1.013}{b^2}\right)^{\frac{1}{2}}} \cdot \frac{(\log N)^\alpha}{N} \quad (5.16)$$

for $N \geq 500$, $1.75 \leq b < 2$, and $N \geq \frac{1}{(1.05)^{\frac{1}{\alpha}} - 1}$. Then, if $b = 1.75$, the error becomes

$$|E_N| \leq 22\gamma \frac{(\log N)^\alpha}{N} \quad (5.17)$$

as desired.

CHAPTER VI

RESULTS

In this chapter, the results from the application of the acceleration method will be discussed for each of the three problems examined.

6.1 First Integral Problem

The first integral was considered in dimensions one, two, four, and six. It is a smooth, continuous exponential function.

$$\int_0^1 \int_0^1 \cdots \int_0^1 \exp(x_1 x_2 \cdots x_s) dx_1 dx_2 \cdots dx_s, \text{ for } s = 1, 2, 4, 6 \quad (6.1)$$

where s is the dimension of the integral. Recall that in one dimension, the Faure and Halton sequences are identical. In the graph below, the quasi-Monte Carlo results are plotted against the accelerated quasi-Monte Carlo results.

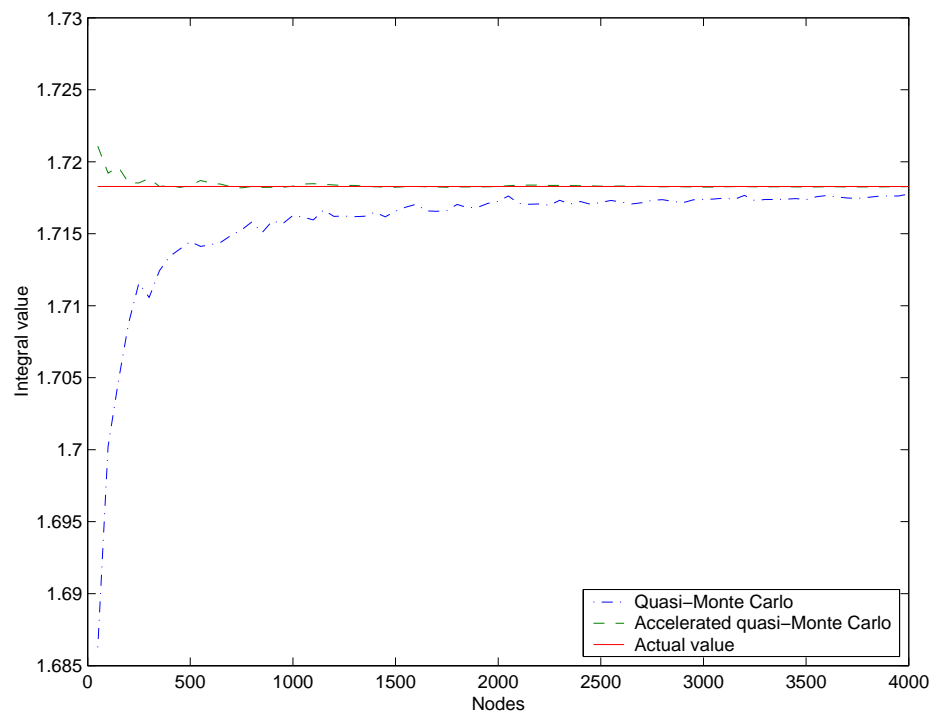


Figure 6.1: First Integral for $s = 1$

Next consider the same integral with $s = 2$. In this case, the Halton and Faure sequences yield different results. Below is a table of these values.

Table 6.1: Results For First Integral in Two Dimensions

Nodes	Halton	Accel. Halton	Faure	Accel. Faure
100	.12868565E+01	.13409771E+01	.12929286E+01	.13422426E+01
200	.13011165E+01	.13271047E+01	.13070381E+01	.13306252E+01
300	.13070934E+01	.13238834E+01	.13090006E+01	.13246650E+01
400	.13094602E+01	.13216083E+01	.13125676E+01	.13214658E+01
500	.13110143E+01	.13207812E+01	.13138231E+01	.13213205E+01
600	.13121791E+01	.13203372E+01	.13125620E+01	.13204254E+01
700	.13129874E+01	.13198520E+01	.13143108E+01	.13200579E+01
800	.13139893E+01	.13197089E+01	.13147753E+01	.13198721E+01
900	.13142257E+01	.13193738E+01	.13157063E+01	.13194582E+01
1000	.13142277E+01	.13191971E+01	.13157579E+01	.13194501E+01
1100	.13143038E+01	.13190311E+01	.13156478E+01	.13192109E+01
1200	.13148235E+01	.13188504E+01	.13157156E+01	.13190507E+01
1300	.13150613E+01	.13187597E+01	.13157318E+01	.13189896E+01
1400	.13152983E+01	.13186550E+01	.13155979E+01	.13187562E+01
1500	.13153920E+01	.13186412E+01	.13159635E+01	.13186964E+01
1600	.13156084E+01	.13186107E+01	.13161687E+01	.13185943E+01
1700	.13159157E+01	.13185562E+01	.13161876E+01	.13184969E+01
1800	.13159778E+01	.13185108E+01	.13160312E+01	.13184431E+01
1900	.13157641E+01	.13184633E+01	.13163702E+01	.13183411E+01
2000	.13160945E+01	.13184577E+01	.13166527E+01	.13183615E+01
2100	.13162822E+01	.13184639E+01	.13167300E+01	.13183692E+01

Actual value is approximately 1.31790.

Table 6.2: Results For First Integral in Two Dimensions, Continued

Nodes	Halton	Accel. Halton	Faure	Accel. Faure
2200	.13162001E+01	.13184772E+01	.13167493E+01	.13183384E+01
2300	.13162458E+01	.13184510E+01	.13167301E+01	.13183273E+01
2400	.13164616E+01	.13184129E+01	.13166651E+01	.13183039E+01
2500	.13162478E+01	.13183757E+01	.13169087E+01	.13182813E+01
2600	.13164989E+01	.13183345E+01	.13167646E+01	.13182801E+01
2700	.13165644E+01	.13183024E+01	.13168081E+01	.13182448E+01
2800	.13163668E+01	.13182626E+01	.13170613E+01	.13182228E+01
2900	.13165265E+01	.13182460E+01	.13168219E+01	.13182013E+01
3000	.13166280E+01	.13182294E+01	.13170341E+01	.13181717E+01
3100	.13165909E+01	.13182151E+01	.13170222E+01	.13181829E+01
3200	.13166407E+01	.13181994E+01	.13170513E+01	.13181653E+01
3300	.13165928E+01	.13181692E+01	.13170081E+01	.13181524E+01
3400	.13167491E+01	.13181539E+01	.13169549E+01	.13181447E+01
3500	.13167161E+01	.13181399E+01	.13169372E+01	.13181276E+01
3600	.13169264E+01	.13181307E+01	.13171488E+01	.13181401E+01
3700	.13168101E+01	.13181201E+01	.13171142E+01	.13181229E+01
3800	.13168024E+01	.13181018E+01	.13171810E+01	.13181226E+01
3900	.13169434E+01	.13181038E+01	.13172423E+01	.13181316E+01
4000	.13168942E+01	.13180971E+01	.13173165E+01	.13181187E+01

Actual value is approximately 1.31790.

According to these results, it appears that the Faure sequence is converging faster than the Halton sequence, while the accelerated Halton sequence is converging faster than the accelerated Faure. Regardless, it is clear that the accelerated sequences are performing much better than the basic sequences. Even with just 2000 nodes, the error is about .00005 for the accelerated Halton sequence. By way of comparison it

would take approximately 20000 nodes to achieve a similar degree of accuracy with standard Monte Carlo.

The following six graphs show the results obtained for this first integral with $s = 2$, $s = 4$, and $s = 6$ for both Halton and Faure sequences. Note that the scale is not the same on each.

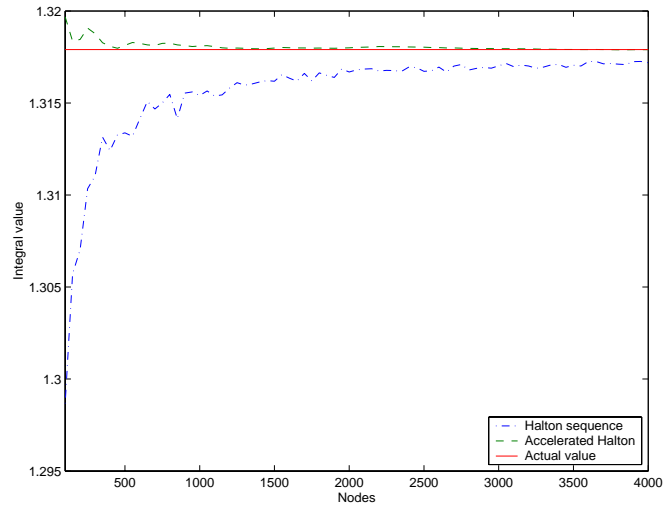


Figure 6.2: First Integral for the Halton Sequence with $s = 2$

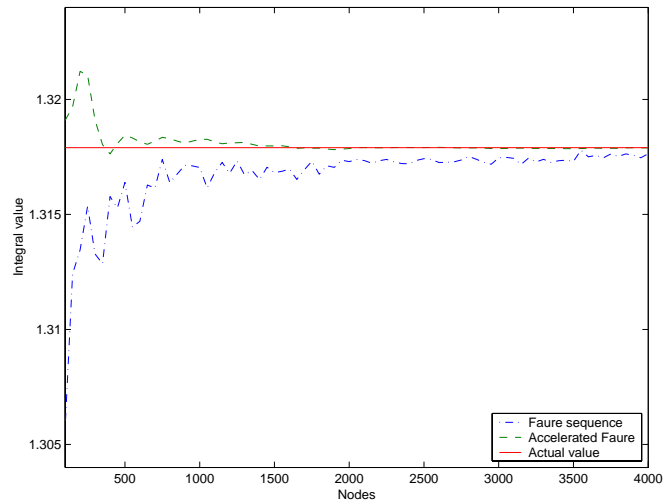


Figure 6.3: First Integral for the Faure Sequence with $s = 2$

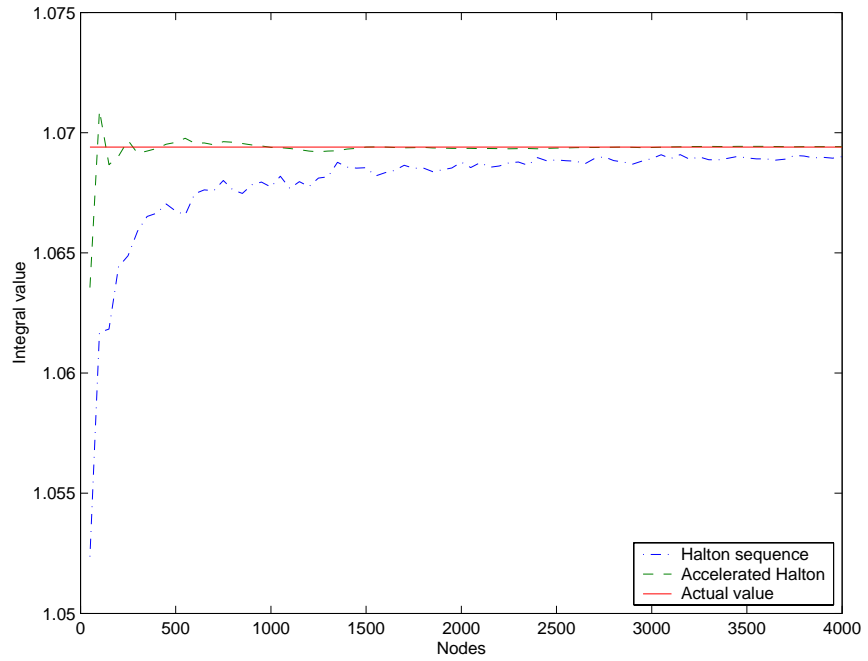


Figure 6.4: First Integral for the Halton Sequence with $s = 4$

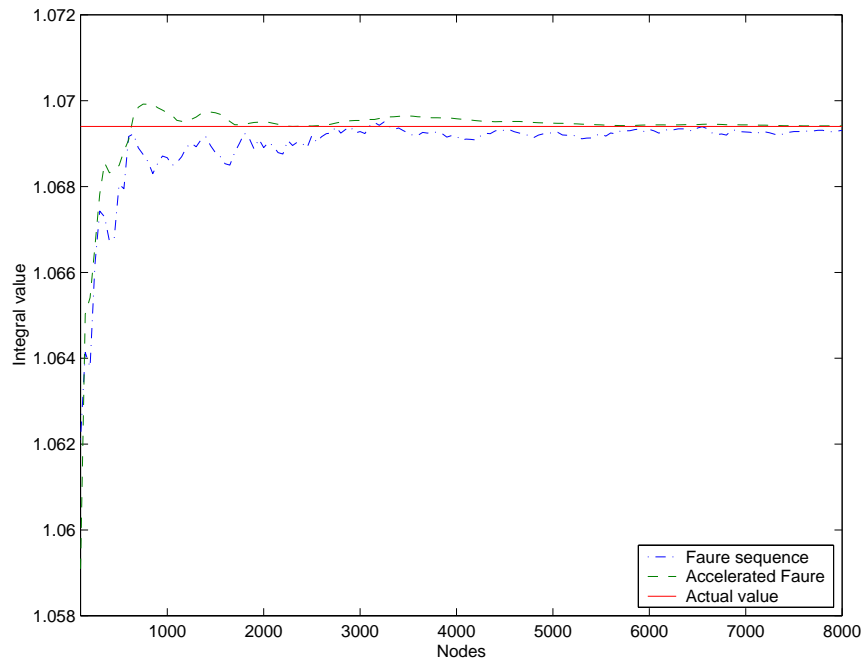


Figure 6.5: First Integral for the Faure Sequence with $s = 4$

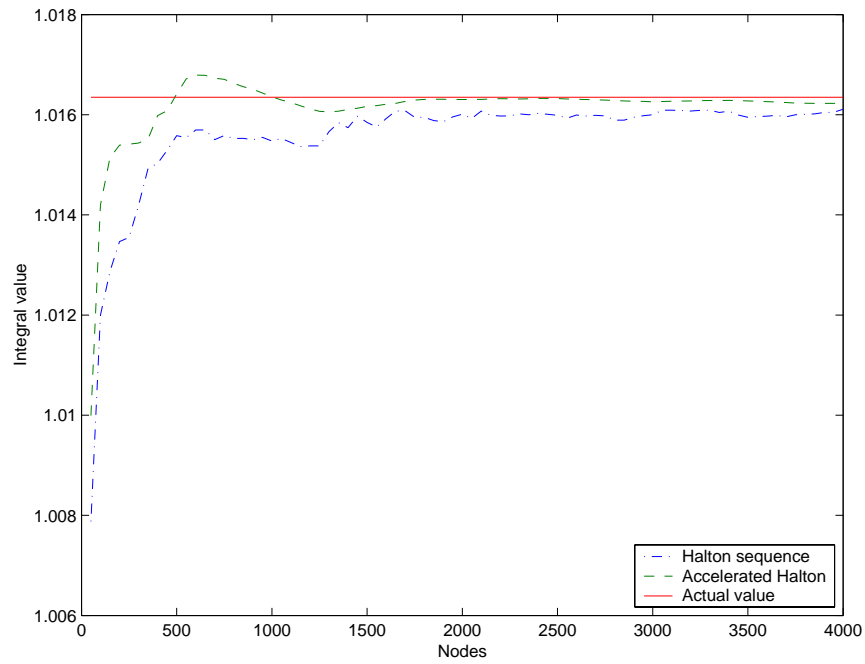


Figure 6.6: First Integral for the Halton Sequence with $s = 6$

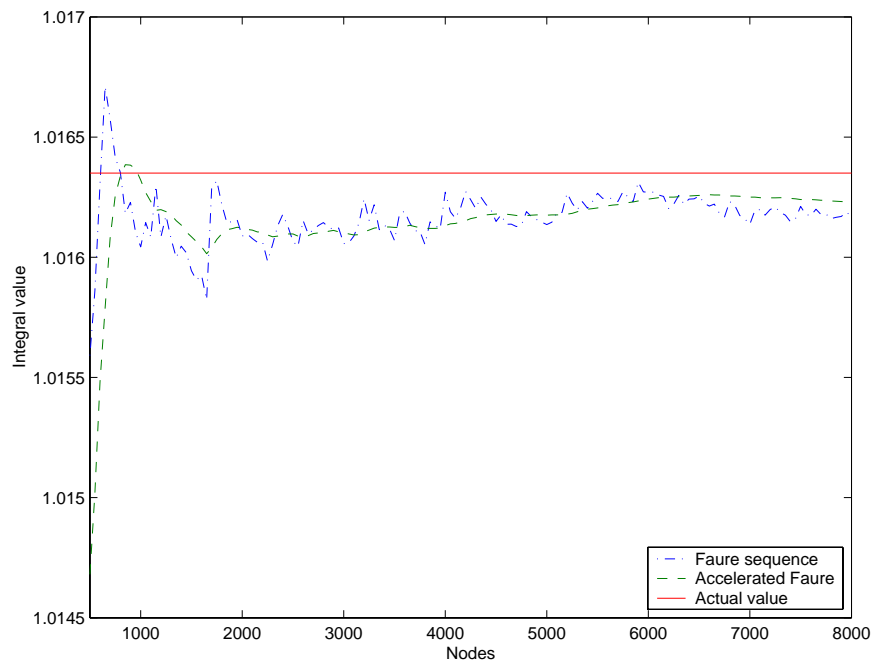


Figure 6.7: First Integral for the Faure Sequence with $s = 6$

In figure 6.7, the graph has been zoomed in to show greater detail. It can be seen how the smoothing procedure works in this image, through making the oscillations less dramatic. For $s = 6$, neither the quasi-Monte Carlo methods nor the accelerated methods appear to be converging as quickly as for the lower dimensions, and the accelerated methods are not a drastic improvement upon the original sequences. This can be explained by the fact that the method is expected to perform best for smoother functions and lower dimensions, as mentioned previously. Thus what was illustrated in figure 6.1 is the best case scenario, with a smooth function integrated over just one dimension. Though the accelerated method is still very effective for $s = 2$ and $s = 4$, its advantage lessens progressively as the dimension increases.

6.2 Second Integral Problem

The second integral studied was also considered in dimensions one, two, four, and six. This particular integral, found in [3], has a discontinuous integrand.

$$\int_0^1 \int_0^1 \cdots \int_0^1 \left(\frac{1}{R_s}\right)^s \chi_s(x_1, x_2, \dots, x_s) dx_1 dx_2 \cdots dx_s \quad (6.2)$$

where

$$R_s = \left(\frac{7}{8}\right)^{\frac{6}{s}}$$

and

$$\chi_s(x_1, x_2, \dots, x_s) = \begin{cases} 1 & \text{if } |x_i - .5| \leq \frac{R_s}{2}, \text{ for } i = 1, 2, \dots \\ 0 & \text{otherwise.} \end{cases}$$

and s is the dimension.

This integral has been constructed so that its value will always be one. Once again, note that the Halton and Faure sequences are identical in one dimension. The results are plotted below.

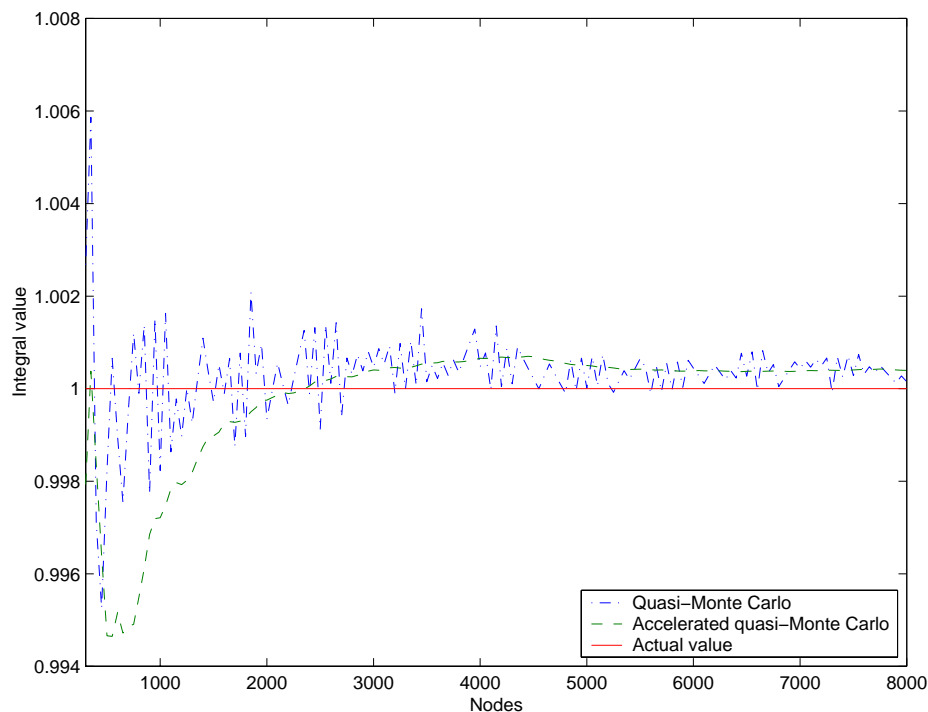


Figure 6.8: First Integral for $s = 1$

Next consider the same problem with $s = 2$. Again, the results for both Halton and Faure sequences have been placed in a table below.

Table 6.3: Results For Second Integral in Two Dimensions

Nodes	Halton	Accel. Halton	Faure	Accel. Faure
100	.98040247E+00	.10455669E+01	.91355681E+00	.86783600E+00
200	.95811951E+00	.99466479E+00	.98040128E+00	.95881593E+00
300	.98782784E+00	.98523450E+00	.99525511E+00	.98047251E+00
400	.10082525E+01	.10073173E+01	.99154115E+00	.99314004E+00
500	.99376911E+00	.10084280E+01	.10026819E+01	.10001397E+01
600	.99897003E+00	.10043088E+01	.99897003E+00	.10033579E+01

Actual value is 1.

Table 6.4: Results For Second Integral in Two Dimensions, Continued

Nodes	Halton	Accel. Halton	Faure	Accel. Faure
700	.10026853E+01	.10061626E+01	.10026853E+01	.10060951E+01
800	.99990118E+00	.10042210E+01	.99711591E+00	.10071231E+01
900	.10002116E+01	.10027409E+01	.10026873E+01	.10084604E+01
1000	.10004599E+01	.10032599E+01	.10049163E+01	.10089742E+01
1100	.10047123E+01	.10031735E+01	.10107892E+01	.10087554E+01
1200	.10026852E+01	.10031875E+01	.99897158E+00	.10091505E+01
1300	.10009698E+01	.10034490E+01	.10026838E+01	.10098289E+01
1400	.99790800E+00	.10021832E+01	.10058657E+01	.10100305E+01
1500	.99822527E+00	.10012451E+01	.10071379E+01	.10099595E+01
1600	.10012881E+01	.10010200E+01	.10040733E+01	.10099034E+01
1700	.10013692E+01	.10013285E+01	.10026798E+01	.10091823E+01
1800	.10002035E+01	.10016668E+01	.10014414E+01	.10082030E+01
1900	.10003331E+01	.10017884E+01	.99916041E+00	.10072591E+01
2000	.10015639E+01	.10018349E+01	.99933577E+00	.10064490E+01
2100	.10058621E+01	.10022914E+01	.10005567E+01	.10054682E+01
2200	.10006552E+01	.10023470E+01	.10016680E+01	.10045000E+01
2300	.10026827E+01	.10022824E+01	.99880743E+00	.10035181E+01
2400	.10045413E+01	.10027103E+01	.99804211E+00	.10026296E+01
2500	.10009032E+01	.10027424E+01	.99733800E+00	.10017040E+01
2600	.10009732E+01	.10025140E+01	.99583101E+00	.10007302E+01
2700	.10002127E+01	.10023700E+01	.99691153E+00	.99992234E+00
2800	.99871069E+00	.10021025E+01	.99711907E+00	.99925631E+00
2900	.10011541E+01	.10018777E+01	.10011541E+01	.99880850E+00

Actual value is 1.

Table 6.5: Results For Second Integral in Two Dimensions, Continued

Nodes	Halton	Accel. Halton	Faure	Accel. Faure
3000	.10012064E+01	.10019950E+01	.99897814E+00	.99872679E+00
3100	.10012553E+01	.10019640E+01	.99981767E+00	.99892658E+00
3200	.10006047E+01	.10019604E+01	.99990845E+00	.99908757E+00
3300	.10006690E+01	.10019282E+01	.10013442E+01	.99919468E+00
3400	.10000739E+01	.10017253E+01	.10007293E+01	.99933994E+00
3500	.10001496E+01	.10014169E+01	.10026963E+01	.99970919E+00
3600	.10008401E+01	.10012732E+01	.10020779E+01	.10002007E+01
3700	.10008910E+01	.10011345E+01	.10026976E+01	.10005423E+01
3800	.10009391E+01	.10010580E+01	.10015255E+01	.10007086E+01
3900	.10015562E+01	.10011338E+01	.10009848E+01	.10008577E+01
4000	.10004712E+01	.10010947E+01	.10010283E+01	.10008792E+01
4500	.99823576E+00	.10006548E+01	.10021967E+01	.10009775E+01
5000	.10004559E+01	.99985147E+00	.10026839E+01	.10015702E+01
5500	.10002468E+01	.99960756E+00	.10055131E+01	.10031689E+01
6000	.10000726E+01	.99984229E+00	.10034146E+01	.10038891E+01
6500	.99958241E+00	.99976295E+00	.10016390E+01	.10036781E+01
7000	.10004354E+01	.99963158E+00	.10007536E+01	.10030026E+01
7500	.10002835E+01	.99948323E+00	.10029571E+01	.10027068E+01
8000	.99987203E+00	.99949223E+00	.10012646E+01	.10024644E+01

Actual value is 1.

According to the above results, the accelerated methods are actually decreasing the accuracy of the approximation. For this particular problem, neither quasi-Monte Carlo sequences nor the accelerated methods worked particularly well. Also, the accelerated methods did not improve the convergence in any noticeable manner. The only real advantage that the accelerated method gives for this particular problem is

that the results form a smoother curve with much less drastic oscillations than before the acceleration. Such a smoothing operation can still be of value if. For instance, if a certain degree of accuracy is required, the accelerated sequences typically will not stray out of that bound once it is reach, while the original sequences continue to oscillate. This can be further illustrated by the following six graphs which plot the results for $s = 2$, $s = 4$, and $s = 6$, for both the Halton and Faure sequences. Again, note that the scale is not the same for each graph.

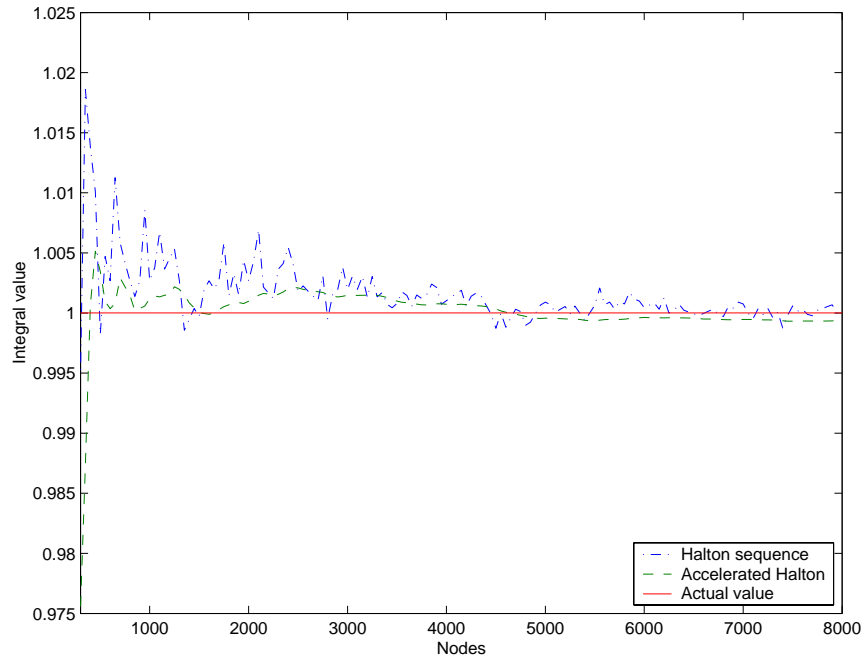


Figure 6.9: Second Integral for the Halton Sequence with $s = 2$

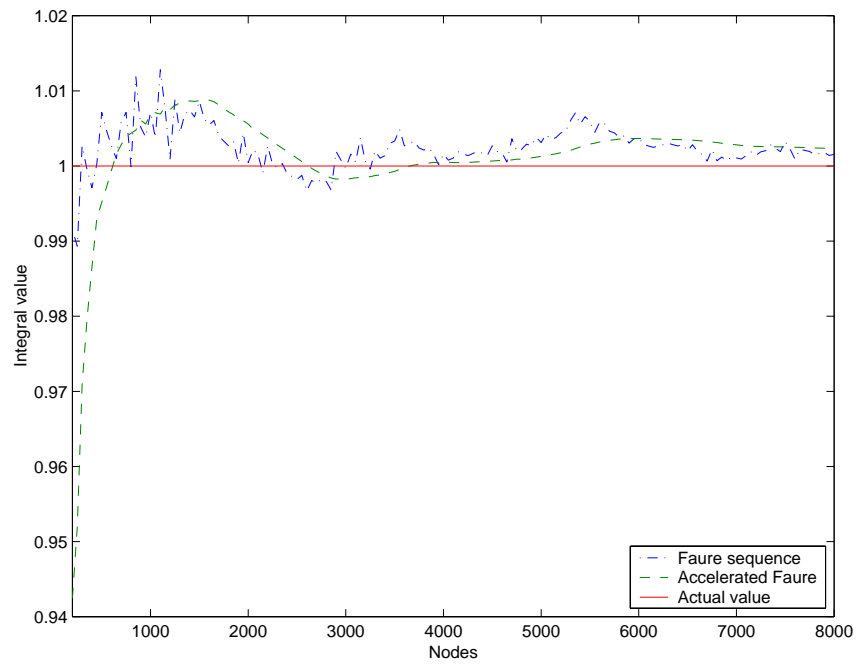


Figure 6.10: Second Integral for the Faure Sequence with $s = 2$

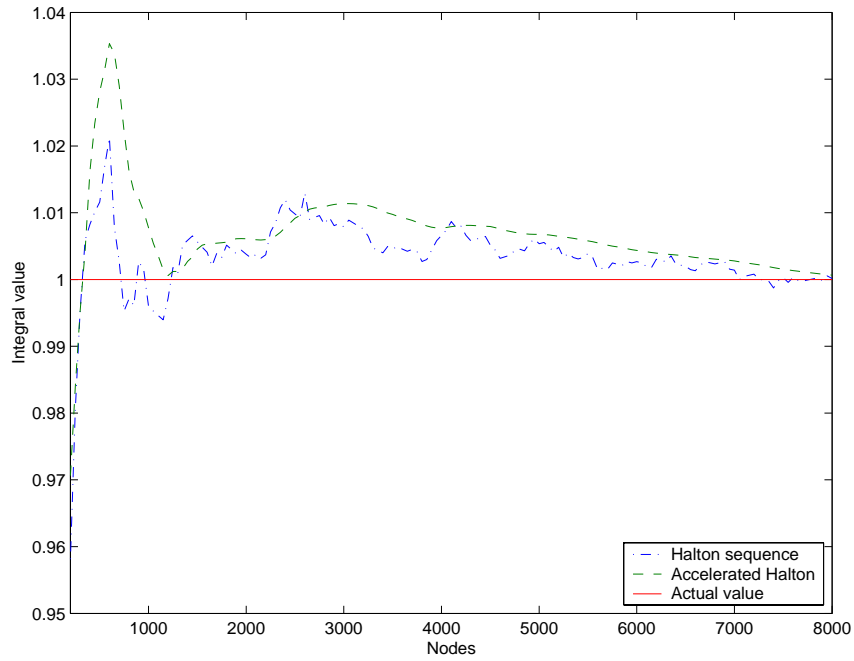


Figure 6.11: Second Integral for the Halton Sequence with $s = 4$

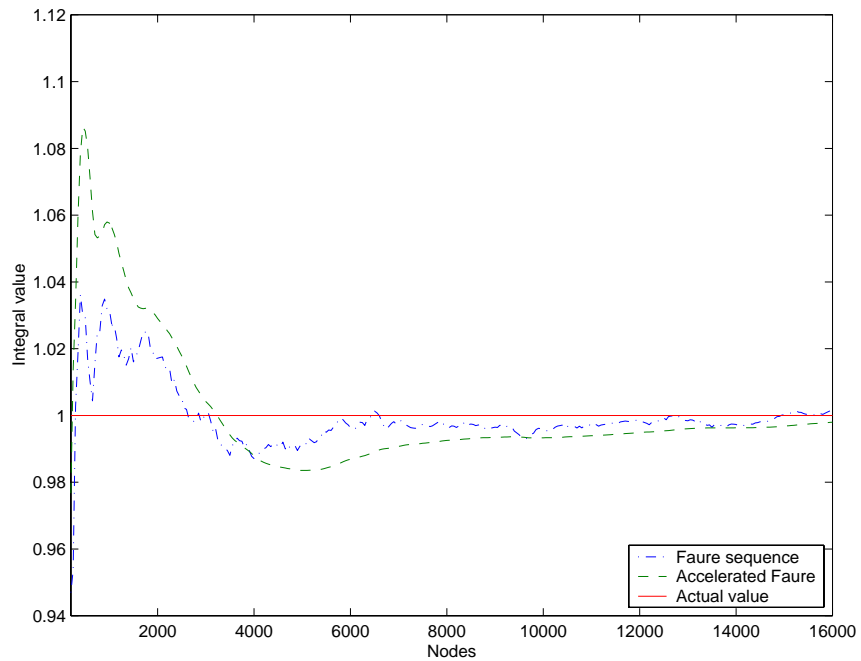


Figure 6.12: Second Integral for the Faure Sequence with $s = 4$

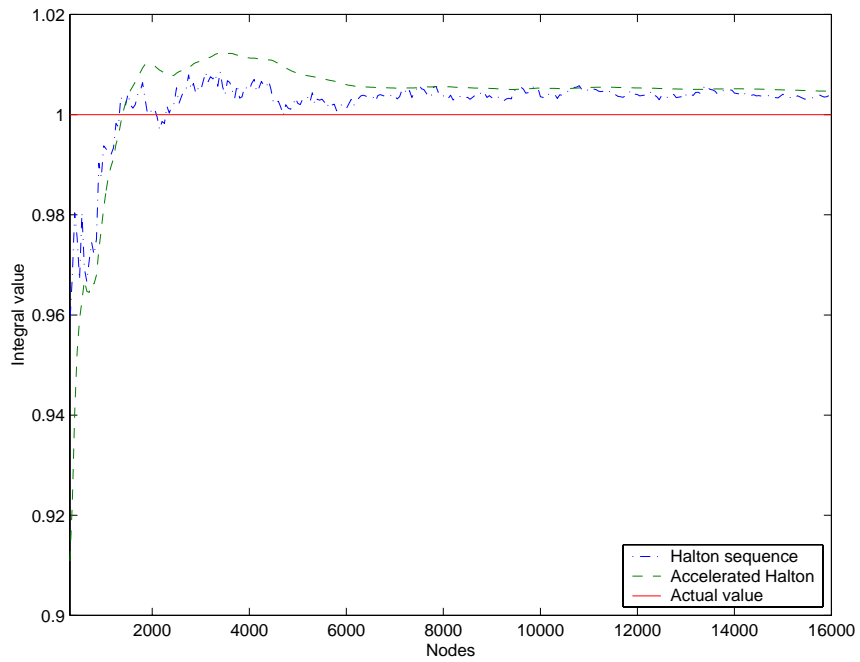


Figure 6.13: Second Integral for the Halton Sequence with $s = 6$

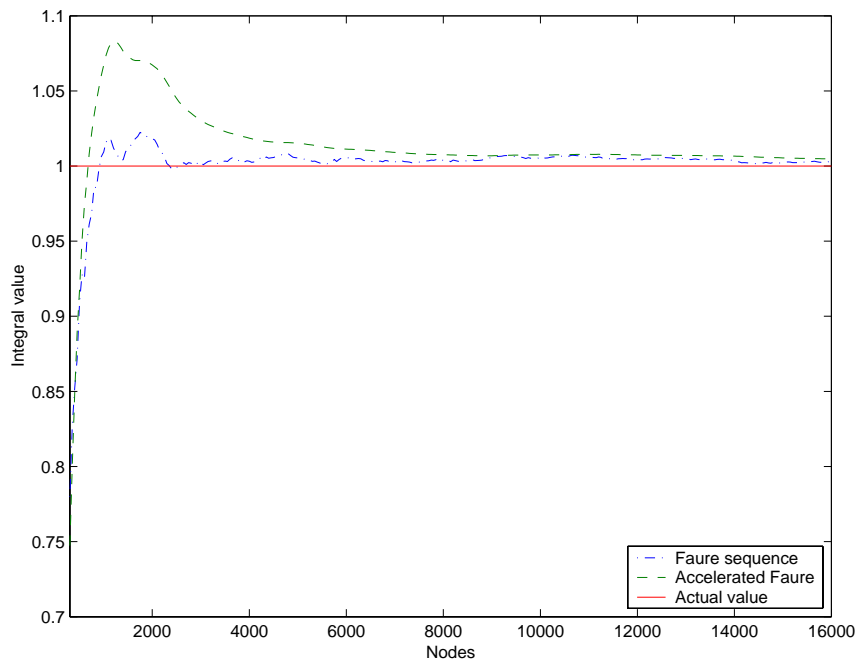


Figure 6.14: Second Integral for the Faure Sequence with $s = 6$

6.3 Third integral Problem

The third integral studied was a European call option, which is strictly a one dimensional problem.

$$C = \frac{e^{-rT}}{\sqrt{2\pi\sigma^2T}} \int_{\ln s}^{\infty} (e^x - S) e^{-\frac{(x - \ln S - rT + \frac{\sigma^2T}{2})}{2\sigma^2T}} dx \quad (6.3)$$

where

$$T = 1, r = 0.1, S = 100, \sigma = 0.3, \text{ and } s = 1$$

and T is the time to maturity, r is the riskless rate, S is the price of the stock, σ is the volatility, s is the dimension, and C is the value of the call. Now let

$$z = \frac{(x - \ln S - rT + \frac{\sigma^2T}{2})}{\sigma\sqrt{T}}$$

and

$$b = \frac{(-rT + \frac{\sigma^2T}{2})}{\sigma\sqrt{T}}$$

Then

$$C = \frac{e^{-rT}}{\sqrt{2\pi}} S \int_b^{\infty} \left[e^{\sigma\sqrt{T}z} e^{rT} e^{-\frac{\sigma^2T}{2}} - 1 \right] e^{-\frac{z^2}{2}} dz \quad (6.4)$$

Now letting

$$h(z) = \frac{1}{2\pi} e^{-\frac{z^2}{2}}$$

and

$$y = \int_{-\infty}^z h(t) dt$$

and

$$a = \int_{-\infty}^b h(t) dt$$

then

$$C = S e^{-rT} \int_a^1 \left[e^{\sigma\sqrt{T}G(y)} e^{rT} e^{-\frac{\sigma^2T}{2}} - 1 \right] dy \quad (6.5)$$

where $G(y) = z$ if

$$y = \int_{-\infty}^z \frac{e^{-\frac{t^2}{2}}}{\sqrt{2\pi}} dt$$

Finally, let

$$w = \frac{y - a}{1 - a}$$

so that $y = (1 - a)w + a$ and

$$C = Se^{-rT} \int_0^1 \left[e^{\sigma\sqrt{T}G((1-a)w+a)} e^{rT} e^{-\frac{\sigma^2 T}{2}} - 1 \right] (1 - a)dw \quad (6.6)$$

For $T = 1$, $r = .1$, $S = 100$, $\sigma = .3$, then

$$C \approx 90.4837 \int_0^1 \left[e^{.3G((1-a)w+a)} (1.05654) - 1 \right] (1 - a)dw \quad (6.7)$$

where

$$a \approx \int_{-\infty}^{-.183333} \frac{e^{-\frac{t^2}{2}}}{\sqrt{2\pi}} dt \approx 0.427268$$

Thus,

$$C \approx 51.8229 \int_0^1 \left[1.05654e^{.3G(.572732w+.427268)} - 1 \right] dw \quad (6.8)$$

where $G : U[0, 1] \rightarrow N[0, 1]$.

According to [2], the value of C is about 16.734. Applying the method of acceleration, the following results are obtained.

Table 6.6: Results For Third Integral

Nodes	Quasi-Monte Carlo	Accelerated
100	.15930358E+02	.16563065E+02
200	.16274723E+02	.16626125E+02
300	.16358376E+02	.16679771E+02
400	.16477859E+02	.16670567E+02
500	.16502121E+02	.16674673E+02
600	.16521236E+02	.16702620E+02
700	.16531742E+02	.16695034E+02
800	.16593212E+02	.16698151E+02

Actual value is approximately 16.734.

Table 6.7: Results For Third Integral, Continued

Nodes	Quasi-Monte Carlo	Accelerated
900	.16603451E+02	.16697300E+02
1000	.16605453E+02	.16701492E+02
1100	.16613737E+02	.16715788E+02
1200	.16615974E+02	.16716263E+02
1300	.16623272E+02	.16715466E+02
1400	.16621325E+02	.16712292E+02
1500	.16624578E+02	.16710833E+02
1600	.16657274E+02	.16714132E+02
1700	.16637730E+02	.16714081E+02
1800	.16661531E+02	.16713886E+02
1900	.16644785E+02	.16714687E+02
2000	.16663923E+02	.16716413E+02
2100	.16671717E+02	.16721632E+02
2200	.16667032E+02	.16724409E+02
2300	.16667629E+02	.16724604E+02
2400	.16669939E+02	.16724775E+02
2500	.16669416E+02	.16723961E+02
2600	.16672764E+02	.16724335E+02
2700	.16669661E+02	.16723833E+02
2800	.16672880E+02	.16722656E+02
2900	.16668087E+02	.16722412E+02
3000	.16673998E+02	.16721884E+02
3100	.16686466E+02	.16722885E+02
3200	.16692852E+02	.16723663E+02

Actual value is approximately 16.734.

Table 6.8: Results For Third Integral, Continued

Nodes	Quasi-Monte Carlo	Accelerated
3300	.16679308E+02	.16723593E+02
3400	.16681177E+02	.16723656E+02
3500	.16677877E+02	.16723309E+02
3600	.16694719E+02	.16723579E+02
3700	.16684029E+02	.16723898E+02
3800	.16685152E+02	.16723955E+02
3900	.16688488E+02	.16724470E+02
4000	.16696539E+02	.16724968E+02
5000	.16698145E+02	.16728260E+02
6000	.16701138E+02	.16727148E+02
7000	.16702860E+02	.16727861E+02
8000	.16713762E+02	.16728971E+02
9000	.16710964E+02	.16730766E+02
10000	.16713837E+02	.16730253E+02
11000	.16712513E+02	.16729927E+02
12000	.16715872E+02	.16729567E+02
13000	.16715704E+02	.16730009E+02
14000	.16716446E+02	.16729828E+02
15000	.16717749E+02	.16730022E+02
16000	.16722597E+02	.16730377E+02

Actual value is approximately 16.734.

It can be seen in the previous table that the accelerated sequence is converging much faster than the quasi-Monte Carlo, and this advantage is particularly obvious when the number of nodes is less than 4000. The following graph further illustrates the data presented in the table above.

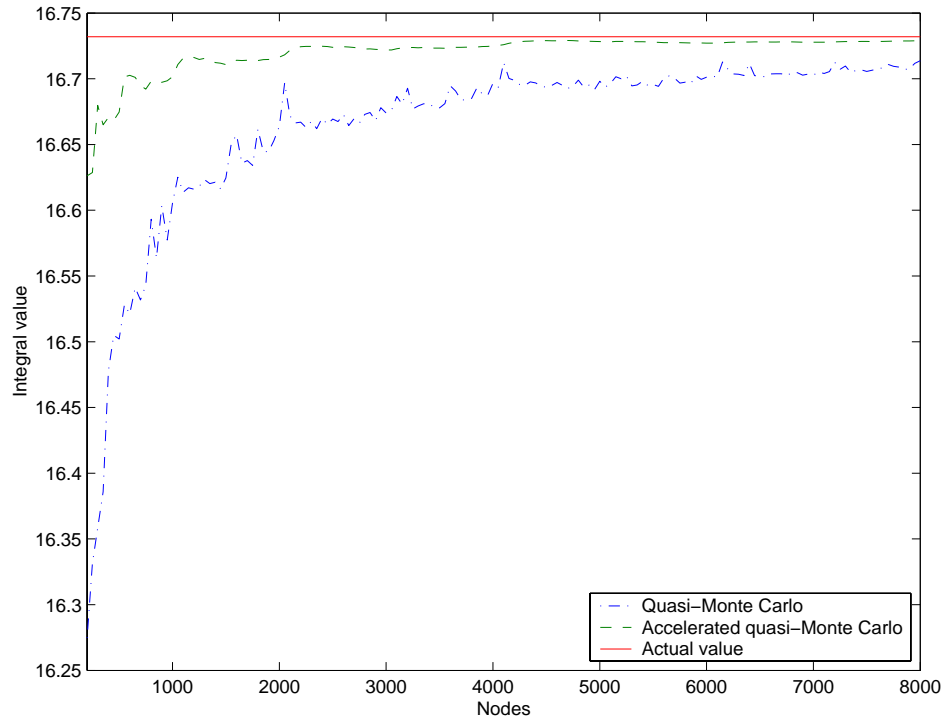


Figure 6.15: Third Integral Results

CHAPTER VII

CONCLUSION

The method of acceleration described in this paper has shown to be effective for one type problem while ineffective for another type. Specifically, the method works well for integrals over smaller dimensions of functions that are smoother. The second integral studied in Chapter VI, the discontinuous integrand problem, showed that this method of acceleration, and perhaps even quasi-Monte Carlo sequences in general, cannot be applied universally. However, the method showed good success for the European call option problem. The method was expected to work well since the call option was a one dimensional integral of an exponential function. Should this method then be applied to similar integrals, it would appear to have an equivalent rate of success. Given that a significant number of integrals in financial mathematics satisfy these criteria of low dimension and smooth integrands, this method should prove to be very useful in that field.

Through this investigation, it was not apparent whether the accelerated Halton or the accelerated Faure sequence performed better than the other. The advantage of Halton, however, is that it seems to be more easily adaptable to parallel computing than the Faure sequence due to the fact that the values of nodes in higher dimensions do not depend on previous dimensions. Also, as illustrated by the figures in Chapter III, the Halton sequence appears to have less regularity and is more evenly distributed.

Future work on this topic could include finding a better error estimate than the one used in this paper. Also, a method of determining when the Halton or Faure sequence would be more effective than the other could be useful.

BIBLIOGRAPHY

- [1] R. Caffisch, W. Morokoff, and A. Owen, "Valuation of Mortgage-Backed Securities Using Brownian Bridges to Reduce Effective Dimension," 301-314, in *Monte Carlo: Methodologies and Applications for Pricing and Risk Management*, Bruno Dupire, ed., Risk Books, London, (1998).
- [2] C. Joy, P. Boyle, and K. Tan, "Quasi-Monte Carlo Methods in Numerical Finance," 269-280, in *Monte Carlo: Methodologies and Applications for Pricing and Risk Management*, Bruno Dupire, ed., Risk Books, London, (1998).
- [3] W. Morokoff and R. Caffisch, "Quasi-Monte Carlo Integration," *Journal of Computational Physics*, **122**, 218-230 (1995).
- [4] H. Niederreiter, *Random Number Generation and Quasi-Monte Carlo Methods*, SIAM, Philadelphia, (1992).

APPENDIX A
PROGRAM QMFAURE1.F

This code, written in Fortran 77, implemented the method of acceleration on the Faure sequence to solve the first integral studied.

```
dimension irr(2),phi(10),dims(4)
data irr/5, 7/
data dims/1, 2, 4, 6/
do 77 iii=1,4
sdim=dims(iii)
write(output,*) 'sdim = ',sdim
mmm=32768
s=0.0
sa=0.0
sb=0.0
sc=0.0
sd=0.0
se=0.0
if (sdim.le.2) then
ir=2
else
ir=irr(iii-2)
end if
n=0
do 300 nn=1,mmm
n=n+1
call faure(n,ir,sdim,phi)
s=s+func(sdim,phi)
if (n.ne.1) then
an=real(n)
x=an/log(an)
w=s/an
y=s/log(an)
sa=sa+y
sb=sb+x
sc=sc+x*y
sd=an-1.0
se=se+x*x
yy=((sa*sb)-(sc*sd))/((sb*sb)-(sd*se))
if(n.le.4000) then
num=mod(n,100)
else
num=mod(n,1000)
endif
if(num.eq.0) then
write(6,150) n,w,yy
150 format(2x,i8,4x,e15.8,4x,e15.8)
end if
end if
300 continue
```

```

77  continue
    stop
    end

    subroutine faure(n,ir,sdim,phi)
    dimension ia(100),aa(10,100),phi(100)
    m=0
311  continue
    m=m+1
    khc=ir**m
    if(khc.le.n) goto 311
    nn=n
    m=m-1
    do 400 j1=1,m+1
    j=m+1-j1
    khc=ir**j
    ia(j+1)=nn/khc
    nn=nn-ia(j+1)*khc
400  continue
    phi(1)=0.0
    do 500 j=1,m+1
500  aa(1,j)=ia(j)
    air=ir
    do 505 j=1,m+1
    aj=j
    chk=1.0/(air**aj)
505  phi(1)=phi(1)+aa(1,j)*chk
    if (sdim.gt.1) then
    do 520 k=2,sdim
    do 515 j=0,m
    sum=0.0
    do 510 i=j,m
    call combo(i,j,ijk)
    sum=sum+ijk*aa(k-1,i+1)
510  continue
    aa(k,j+1)=mod(sum,air)
515  continue
520  continue
    do 530 k=2,sdim
    phi(k)=0.0
    do 525 j=0,m
    jj=-j-1
    chkk=air**jj
    phi(k)=phi(k)+aa(k,j+1)*chkk
525  continue
530  continue
    end if
    return
    end

    subroutine combo(i,j,ijk)
    call fact(i,ii)

```

```

call fact(j,jj)
k=i-j
call fact(k,kk)
ijk=ii/(jj*kk)
return
end

subroutine fact(i,j)
j=1
if (i .ge. 1) then
do 600 k=1,i
600 j=j*k
end if
return
end

function func(sdim,phi)
dimension phi(10)
vnum=1.0
do 700 k=1,sdim
700 vnum=vnum*phi(k)
continue
func=exp(vnum)
return
end

```

APPENDIX B
PROGRAM QMHALTON2.F

This code, written in Fortran 77, implemented the method of acceleration on the Halton sequence to solve the second integral studied.

```

dimension irr(6),phi(10),dims(4)
data irr/2, 3, 5, 7, 11, 13/
data dims/1, 2, 4, 6/
do 77 iii=1,4
sdim=dims(iii)
write(output,*) 'sdim = ',sdim
mmm=32768
s=0.0
sa=0.0
sb=0.0
sc=0.0
sd=0.0
se=0.0
n=0
do 300 nn=1,mmm
n=n+1
do 289 ir=1,sdim
irrs=irr(ir)
call faure(n,ir,phi,irrs)
289 continue
s=s+func(sdim,phi)
if (n.ne.1) then
an=real(n)
x=an/log(an)
w=s/an
y=s/log(an)
sa=sa+y
sb=sb+x
sc=sc+x*y
sd=an-1.0
se=se+x*x
yy=((sa*sb)-(sc*sd))/((sb*sb)-(sd*se))
num=mod(n,2000)
if(num.eq.0) then
write(6,150) n,y,w,yy
150 format(2x,i8,4x,e12.5,4x,e15.8,4x,e15.8)
end if
end if
300 continue
77 continue
stop
end

subroutine faure(n,ir,phi,irrs)
dimension ia(100),aa(100,10),phi(100)
m=0

```

```

311 continue
    m=m+1
    khc=irrs**m
    if(khc.le.n) goto 311
    nn=n
    m=m-1
    do 400 j1=1,m+1
        j=m+1-j1
        khc=irrs**j
        ia(j+1)=nn/khc
        nn=nn-ia(j+1)*khc
400 continue
    phi(ir)=0.0
    do 500 j=1,m+1
500 aa(j,1)=ia(j)
        air=irrs
        do 505 j=1,m+1
            aj=j
            chk=1.0/(air**aj)
505 phi(ir)=phi(ir)+aa(j,1)*chk
        return
    end

    function func(sdim,phi)
    dimension phi(10)
    func=0.0
    chk=0
    ss=real(sdim)
    rs=(7.0/8.0)**(6.0/ss)
    do 700 k=1,sdim
        vnum=2*abs(phi(k)-.5)
        if (vnum.gt.rs) then
            chk=chk+1
        end if
700 continue
    if (chk.eq.0) then
        func=1.0
    end if
    func = func*(rs**(-ss))
    return
    end

```

APPENDIX C
PROGRAM QMPROB3.F

This code, written in Fortran 77, implemented the method of acceleration on the quasi-Monte Carlo sequence to solve the third integral studied.

```

      sdim=1
      mmm=32768
      s=0.0
      sa=0.0
      sb=0.0
      sc=0.0
      sd=0.0
      se=0.0
      ir=2
      n=0
      do 300 nn=1,mmm
      n=n+1
      call faure(n,ir,phi)
      s=s+func(phi)
      if (n.ne.1) then
      an=real(n)
      x=an/log(an)
      w=s/an
      y=s/log(an)
      sa=sa+y
      sb=sb+x
      sc=sc+x*y
      sd=an-1.0
      se=se+x*x
      yy=((sa*sb)-(sc*sd))/((sb*sb)-(sd*se))
      if(n.le.4000) then
      num=mod(n,200)
      else
      num=mod(n,2000)
      endif
      if(num.eq.0) then
      write(6,150) n,y,w,yy
150  format(2x,i8,4x,e12.5,4x,e15.8,4x,e15.8)
      end if
      end if
300  continue
77   continue
      stop
      end

      subroutine faure(n,ir,phi)
      dimension ia(100),aa(10,100),phi(100)
      m=0
311  continue
      m=m+1
      khc=ir**m
```

```

    if(khc.le.n) goto 311
    nn=n
    m=m-1
    do 400 j1=1,m+1
    j=m+1-j1
    khc=ir**j
    ia(j+1)=nn/khc
    nn=nn-ia(j+1)*khc
400  continue
    phi(1)=0.0
    do 500 j=1,m+1
500  aa(1,j)=ia(j)
    air=ir
    do 505 j=1,m+1
    aj=j
    chk=1.0/(air**aj)
505  phi(1)=phi(1)+aa(1,j)*chk
    return
    end

subroutine uniftonorm(un,ga)
dimension xxx(1001),yyy(1001)
common xxx,yyy
call maketab
call untoga(un,ga)
return
end

subroutine maketab
dimension xxx(1001), yyy(1001)
common xxx,yyy
cc=sqrt(2.0*3.141592654)
xxx(1)=0.0
yyy(1)=0.0
h=.0005
do 10 i=2,1001
xxx(i)=xxx(i-1)+h
hlp=h*cc*exp(.5*yyy(i-1)*yyy(i-1))
hlp2=hlp*(.125+.25*yyy(i-1)*yyy(i-1))
yyy(i)=yyy(i-1)+hlp*(1.0+hlp*(.50*yyy(i-1)+hlp2))
10  continue
return
end

subroutine untoga(un,ga)
dimension xxx(1001),yyy(1001)
common xxx,yyy
unnn=abs(un-.5)
m=(unnn/.0005)+1
hlp=yyy(m)*(xxx(m+1)-unnn) + yyy(m+1)*(unnn-xxx(m))
ga=hlp/.0005
if(un.lt.0.5) ga=-ga

```

```
return
end

function func(phi)
un=(.572732*phi) + .427268
call uniftonorm(un,ga)
func = 51.8229*((1.05654*exp(.3*ga))-1)
return
end
```