Estimating The Effective Dimension Of High-Dimensional Finance Problems Using Sobol' Sensitivity Indices

by

Sara Zgheib

A research paper presented to the
University of Waterloo
In partial fulfillment of the requirements for the degree of
Master of Mathematics
in
Computational Mathematics

supervisor: **Christiane Lemiuex**

Waterloo, Ontario, Canada, 2009

I hereby declare that I am the sole author of this thesis. This is a true copy of the research paper, including any required final revisions, as accepted by my examiners.

I understand that my research paper may be made electronically available to the public.

Abstract

Problems in many disciplines, such as physics, chemistry, and finance, can be modelled as integrals of high dimensions (hundreds or even thousands). Quasi-Monte Carlo (QMC) methods, which perform sampling using a more uniform point set than that used in MC, have been successfully used to approximate multivariate integrals with an error bound of size $O((logN)^k N^{-1})$ or even $O((logN)^k N^{-3/2})$, where N is the size of the sample and k depends on the dimension of the problem. This suggests an outperformance over the standard MC whose error bound is only $O(N^{-1/2})$. But for high dimensional problems, this outperfomance might not appear at feasible sample sizes due to the dependence of the QMC convergence rate on the dimension of the problem. However, around 1993, it was found by researchers at Columbia University that QMC provides better convergence rates than MC for very high-dimensional problems in finance and in physics as well. This may be explained by the fact that the integrands in these problems have low "effective dimension" properties that interact positively with the properties of the point set used by the QMC method. To understand the efficiency of QMC, this paper uses Sobol' method for global sensitivity analysis to investigate features of specific finance problems, digital option pricing and mortgage-backed securities, in dimensions as high as 360. Using Sobol' sequences, we estimate the low-order Sobol' sensitivity indices of these problems and estimate their effective dimensions accordingly. We also examine the efficiency of the Brownian Bridge technique in reducing the estimated effective dimension.

Acknowledgements

I would like to thank my research supervisor Christiane Lemieux who helped making this paper possible.

Contents

Chapter 1

Introduction

Simulation is often the only effective numerical tool for problems that do not exhibit a closed form solution, such as the valuation of entities that have inherent stochastic components. Monte Carlo (MC) simulation is a widely used computational tool in many disciplines of physical science, and more recently in finance, thanks to its flexibility and robustness. MC simulation is an attractive method for the approximation of complex, path dependent derivative securities, i.e. securities whose value depends on the whole trajectory of another security or variable, such as Mortgage Backed Securities (MBS). However, one of the main drawbacks of the standard MC simulation using pseudo-random sequences is that its convergence rate is only $O(N^{-1/2})$ for N sample paths.

To improve upon the slow convergence of standard MC methods, Quasi-Monte Carlo (QMC) methods have been extensively applied in financial problems, where sampling is performed using a point set that exhibits higher uniformity properties than the standard MC pseudo-random point set. In 1995, Paskov and Traub [24] provided numerical experiments to demonstrate the outperformance of QMC methods on a mortgage-backed security problem of dimension 360. This was a surprising result back then since QMC was claimed to be superior to MC only on low-dimensional problems (no more than 20 dimensions). This was considered the case because of the dependence of k in the QMC error bound $O((logN)^k N^{-1})$ on the dimension s, implying that in high dimensions, accurate results cannot be achieved at computationally feasible sample sizes. In other words, N has to grow exponentially with s which renders it impractical in high dimensions.

Despite what the error bound suggets, quasi-Monte Carlo simulation outperforms standard Monte Carlo in many high-dimensional problems using relevant sample sizes. This has drawn the attention of researchers to the features of the integrand estimated in these problems. The estimated high-dimensional integrands exhibit some favorable properties that make them amenable to QMC simulation. These properties can be summarized by the *effective dimension* of these integrands. In fact, Caflisch, Morokoff, and Owen (1997)

suggest that the integrand in the MBS problem could be approximated by a sum of one dimensional terms.

The purpose of this paper is to examine the effective dimension of specific finance problems, such as digital option pricing and mortgage-backed security, using a recenlty proposed method by Sobol'(2001). Using the Sobol' sensitivity indices method, we will estimate the effective dimension of these problems and compare the performance of QMC to that of MC. The remainder of this paper is organized as follows: in chapter 2, we discuss the MC and QMC methods respectively. In chapter 3, we present the ANOVA decomposition and the Sobol' sensitivity indices. Chapter 4 defines the notion of the effective dimension and the Brownian bridge technique for reducing it. In chapter 5, we introduce randomized QMC sequences then apply the discussed concepts on specific finance problems, digital option and mortgage-backed securities valuation. Finally, we present some concluding comments and ideas about future research in chapter 6.

Chapter 2

Quasi-Monte Carlo Versus Monte Carlo Integration Methods

2.1 Monte Carlo Method

Let's first give an example of using MC integration method to approximate the value of a path dependent security such as an MBS. To approximate an MBS value, we have to generate a set of N independetly sampled interest rate paths, aggregate the discounted cash flows of the MBS under each sample path to obtain the value of the MBS and average the aggregated cash flows over the sample paths. For a detailed study of MC methods see Owen (1998). Now let's present this method in general:

Consider $f \in L_2$ defined over $D = [0,1]^s$, where L_2 is the field of square-integrable functions. The integral $I = \int_D f(u)du$ is approximated by:

$$\hat{I}_N = \frac{1}{N}\sum_{i=1}^{N} f(U_i)$$

This approximation results in an error defined as:

$$e_N(f) = \hat{I}_N - I = \frac{1}{N}\sum_{i=1}^{N} f(U_i) - \int_D f(u)du$$

The standard MC sampling method uses U_i drawn from the uniform distribution over the domain $D = [0,1]^s$. Using the law of large numbers we have:

$$P(\lim_{N\to\infty} \hat{I}_N = I) = 1$$

which ensures that Monte Carlo method always converges to the exact value for very large N. Having $f \in L_2$ implies that its variance is finite:

$$\sigma^2 = \int_D (f(u) - I)^2 du < \infty$$

then we have that $e_N(f)$ has mean zero and variance $\frac{\sigma^2}{N}$, which implies that the error bound for MC integration method is $O(N^{-1/2})$.

From the above error bound, it seems that MC method seems to have a slow convergence rate. This slow convergence is explained by the clumping of points in the pseudo random sequence used to generate sample paths. Since the sampled points are chosen independently of one another, it's possible that the chosen points end up landing close to each other. However, MC method still has the advantage that its order of convergence is independent of the problem dimension unlike other integration methods, such as Newton Cotes or Gaussian quadrature methods. The latter methods suffer from the *curse of dimensionality*, as described by Richard Bellman [3] to refer to the fact that the sample size N has to increase significantly, often to an infeasible size, as the dimension increases.

2.2 Quasi-Monte Carlo Methods

The Quasi-Monte Carlo method has better convergence properties than that of the Monte Carlo method due to the uniformity of the quasi-random sequences that fill the space D avoiding the clustering seen in MC sampling. To illustrate this idea, we show in figure (2.1) four plots, each is a two-dimensional projection of 4096 points. These points are either sampled from a Sobol' sequence (the two plots in the top) or a pseudo-random sequence (the two plots in the bottom). Notice the difference in the way points are arranged between the top and the bottom plots. In the pseudo-random sequence, uniformity of points is limited by their clumping. On the other hand, the way quasi-random sequence is constructed (as Sobol' in this case) prevents them from clustering. For a detailed survey of low discrepancy sequences and their application in finance see Paskov (1997), and Caflisch, Morokoff, and Owen (1997).

We can measure the uniformity of a sequence of points in the s-dimensional unit cube $D = [0,1]^s$ using a numerical measure called *discrepancy*. Following the procedure outlined in [14], let's define the star discrepancy which is the most commonly used measure. Let $v = (v_1, v_2, ..., v_s) \in [0,1)^s$ and let's consider all sets of the from:

$$B(v) = \{u \in [0,1)^s \colon 0 \leq u_j \leq v_j, 1 \leq j \leq s\}$$

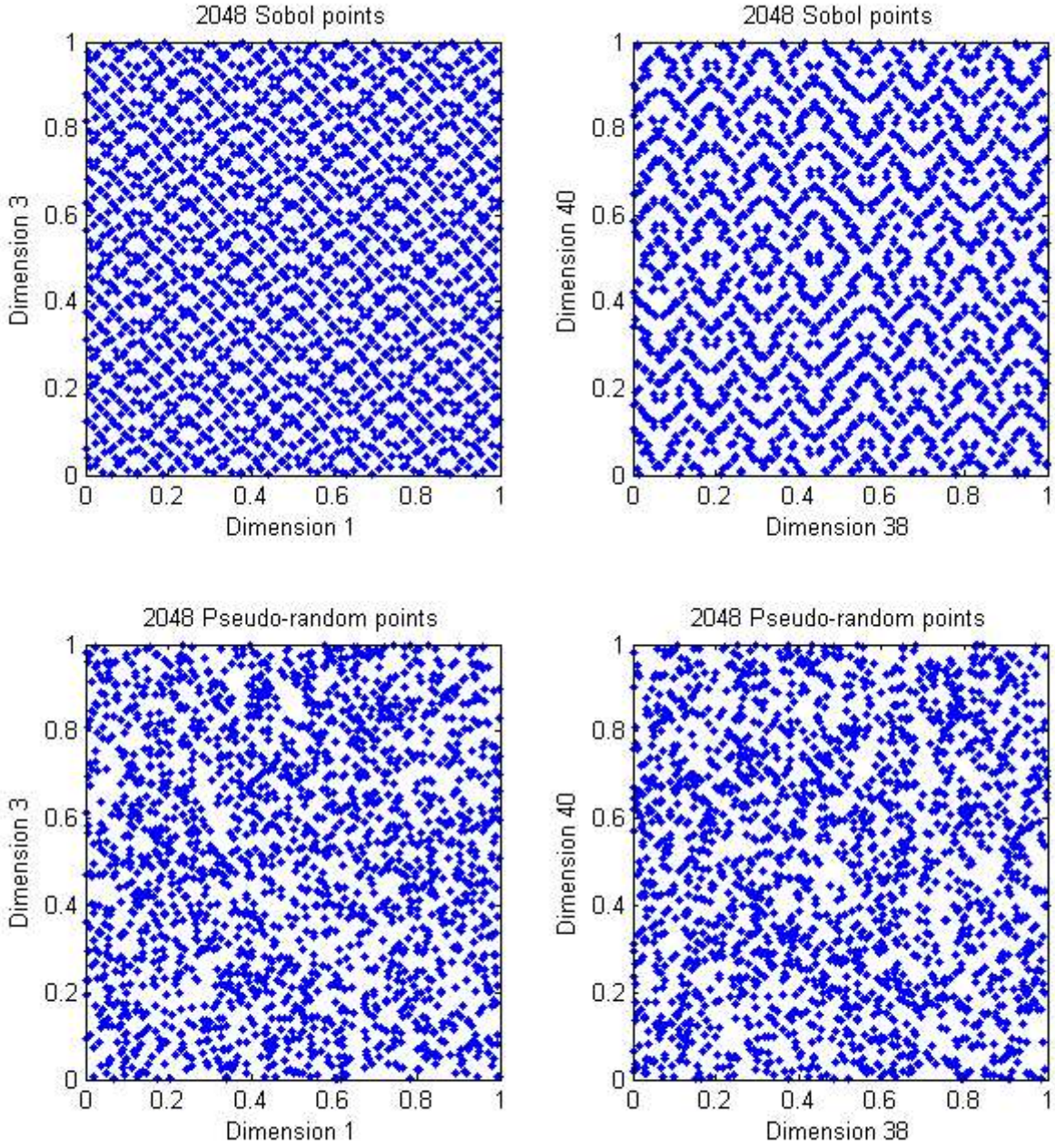
2048 Sobol points
Dimension 3
Dimension 1
2048 Sobol points
Dimension 40
Dimension 38
2048 Pseudo-random points
Dimension 3
Dimension 1
2048 Pseudo-random points
Dimension 40
Dimension 38

Figure 2.1:

Such sets can be thought of as hyper-rectangles with a corner at the origin. Now given an N-point set P_N, we denote by $\alpha(P_N, v)$ the number of elements in P_N that fall in the "box" $B(v)$, i.e. $\alpha(P_N, v)$ denotes the cardinality of the set :

$$\{u_i : 0 \leq u_{i,j} \leq v_j, i = 1, ..., N\}$$

The empirical distribution induced by P_N assigns a probability of $\alpha(P_N, v)/N$ to this box, instead of the value $\prod_{i=1}^{s} v_i$ assigned by the uniform distribution over $[0, 1)^s$. Hence, to measure the departure (or discrepancy) of P_N from uniformity, we compare $\alpha(P_N, v)/N$ to $\prod_{i=1}^{s} v_i$ via the Kolmogorov-Smirnov statistic as follows:

$$D^*(P_N) = \sup_{v \in [0,1)^s} |v_1 v_2 ... v_s - \alpha(P_N, v)/N|$$

Thus, we obtain what we call the star discrepancy measure by maximizing the difference between the two entities. For other definitions of discrepancy and more recent generalizations, we refer the reader to Niedereiter [18] and Hickernell [10] respectively.

The importance of the discrepancy measure can be seen from the Koksma-Hlawka inequality which gives a bound for the error of the simulation-based estimate of the integral of f on $[0, 1)^s$. Following the notation of [5], the Koksma-Hlawka inequality is the following:

$$|e_N(f)| = |1/N \sum_{i=1}^{N} f(u_i) - \int_{[0,1]^s} f(u)du| \leq V(f)D^*(P_N) \tag{2.1}$$

where $V(f)$ is the variation of f. In one dimension, $V(f) = \int_0^1 |df|$. In higher dimensions, the definition becomes much more complicated. Define the quantity:

$$V^{(k)}(f; i_1, ..., i_k) = \int_{I^k} |\frac{\partial^k f}{\partial t_{i_1} ... \partial t_{i_k}}|_{t_j = 1, j \neq i_1, ..., i_k} dt_{i_1} ... dt_{i_k}$$

for all $k \leq s$ and all sets of k integers $1 \leq i_1 < ... < i_k \leq s$. Using the above quantity, we define the variation of f as:

$$V(f) = \sum_{k=1}^{s} \sum_{1 \leq i_1 < ... < i_k \leq s} V^{(k)}(f; i_1, ..., i_k)$$

The Koksma-Hlawka inequality (2.1) should be compared with the formula for the root-mean square error of Monte Carlo integration using a pseudo-random sequence. If $\{u_i\}_{i=1}^{N}$ is an i.i.d. uniform random sample on $D = [0, 1]^s$, then

$$E[e_N(f)^2]^{1/2} = \sigma(f)N^{-1/2} \tag{2.2}$$

where $\sigma(f)$ is the square root of the variance of f given by:

$$\sigma(f) = (\int_{[0,1]^s} [f(u) - E(f)]^2 du)^{1/2}$$

The error magnitudes (2.1) and (2.2) share in common the fact that their bound is a product of two terms, one depends on properties of the integrand function and the other depends on properties of the sequence. Being an absolute bound, the Koksma-Hlawka inequality (2.1) is more reliable than (2.2) which holds only probabilistically. However, the preference is reversed for practical reasons. It is incredibly hard to compute factors in (2.1), but it is much easier to estimate the Monte Carlo variance using the same data needed to estimate $E(f)$. The infinite sequence $\{u_N\}_{i=1}^{\infty}$ is said to be *quasi-random* if:

$$D^*(P_N) \leq c(logN)^k N^{-1}$$

in which the constant c and the logarithmic exponent k are dependent on the dimension s. The Koksma-Hlawka inequality implies that the integration using quasi-random sequences has an error bound of size $O((logN)^k N^{-1})$, which for large enough N makes it much more accurate than standard MC simulation. Examples of quasi-random sequences have been constructed by Halton, Faure, Sobol', Niederreiter, and others. For a detailed discussion, we refer the reader to the monograph of Niederreiter [18].

Although s derivatives of f are required to obtain the variation of f, it was found in practice that quasi-Monte Carlo integration is effective only with a minimal amount of smoothness of f, unless f is discontinuous, where improvements in this case are diminished. To investigate more about the effectiveness of QMC in high-dimensional problems, the reader is referred to [15] and [16]. In these papers, it is shown that the faster convergence rate for QMC generally disappears for high-dimensional problems. A simple evidence for this conclusion is the dependence of the discrepancy on N for different values of the dimension s, where for large s, the discrepancy becomes $O(N^{-1/2})$ same as that of a random sequence, taking on a rate of $O(N^{-1})$ only when N becomes very large. On the other hand, there was almost no problems found for which QMC behaves worse than standard MC.

We have presented above a measure for the uniformity of a sequence, but in practice it's difficult to evaluate the uniformity of a sequence in a high-dimensional space. A necessary but not sufficient condition for uniformity is the uniformity of low-dimensional coordinate projections of the sequence. The Sobol' sequence used in the numerical experiments of this paper has excellent one-dimensional projections and many of its two dimensional ones are highly uniform thanks to the proper choice of the direction numbers as discussed in the section below.

2.2.1 Sobol' Sequence

Sobol' sequence is the most widely used construction in the family of digital sequences and the first one proposed as well. In what follows, we will adopt the same notations and discussion used by C. Lemieux [13] to explain Sobol's sequences construction. For each coordinate j, the one-dimensional projection $P_N(\{j\})$ of P_N requires two things; the first is a primitive polynomial $f_j(z)$ in $\mathbb{F}_2$, the finite field with two elements, and the second one is an integer m_j to initialize a recurrence based on $f_j(z)$ that generates the *direction numbers* defining $P_N(\{j\})$. The method specifies that $f_j(z)$ should be the j^{th} one in the list of primitive polynomials sorted by increasing degree. Sobol' specifies a certain order within each degree as given in the code of Bratley and Fox [4] for $j \leq 40$. The Sobol' sequence generated in our numerical experiments is generated from the package developed by C. Lemieux [13] where the order used for $j > 40$ is that given in the list of primitive polynomial that can be found at Florent Chabaud's website (fchabaud.free.fe).

Assume $f_j(z) = z^q + a_{j,1}z^{q-1} + ... + a_{j,q}$, where $a_{j,l} \in \mathbb{F}_2$ for each j, l. The direction numbers $v_{j,1}, v_{j,2}, ...$ are rationals of the form

$$v_{j,k} = \frac{m_{j,k}}{2^k} = \sum_{l=1}^{k} v_{j,k,l} 2^{-l},$$

where $m_{j,k}$ is an odd integer smaller than 2^k. The first q values $v_{j,1}, v_{j,2}, ..., v_{j,q}$, or equivalently, $m_{j,1}, m_{j,2}, ..., m_{j,q}$, have to be chosen, and the next values follow recursively according to the recurrence relation:

$$v_{j,k} = a_{j,1}v_{j,k-1} \oplus ... \oplus a_{j,q-1}v_{j,k-q+1} \oplus v_{j,k-q} \oplus (v_{j,k-q}/2^q)$$

where $\oplus$ denotes a bit-by-bit exclusive-or operation, and $v_{j,k-q}/2^q$ means that the binary expansion of $v_{j,k-q}$ is shifted by q positions to the right. These direction numbers are then used to define $P_N(\{j\}) = \{u_{i,j}, i = 0, ..., N-1\}$ as follows:

$$u_{i,j} = i_0 v_{j,1} \oplus i_1 v_{j,2} \oplus ... \oplus i_{d-1} v_{j,d} \tag{2.3}$$

where $i_0, ..., i_{d-1}$ are the coefficients in the binary expansion of i, i.e.

$$i = \sum_{l=0}^{d-1} i_l 2^l,$$

and d is such that $N = 2^d$. An alternative way to describe this procedure in correspondance with the general construction principles for digital nets is to consider the binary expansion of $u_{i,j}$ given by:

$$u_{i,j} = \sum_{l=1}^{L} u_{i,j,l} 2^{-L}.$$

Then we have that (2.3) is equivalent to

$$\mathbf{C}^j \begin{bmatrix} i_0 \\ i_1 \\ \vdots \\ i_{d-1} \end{bmatrix} = \begin{bmatrix} u_{i,j,1} \\ u_{i,j,2} \\ \vdots \\ u_{i,j,l} \end{bmatrix}, \tag{2.4}$$

where $\mathbf{C}^j$ is a $L \times d$ matrix defined by

$$\mathbf{C}^j_{l,k} = v_{j,k,l}$$

and the operations in (2.4) are performed in $\mathbb{Z}_2$.

In the implementation of Bratley and Fox [4], the initial values of $m_{j,k}$ for $j \leq 40$ are provided. For dimension $40 < j \leq 360$, C. Lemieux adopts a criterion based on the resolution of P_N for searching for "good" initial values, where resolution is a measure that is often used to assess the quality of pseudo-random number generators based on linear recurrences modulo 2. In Lemieux's package [13], the intial values $m_{j,k}$ for dimension j were chosen as follows: the resolution of $P_N(\{j-i, j\})$ for $i = 1, ..., 8$ and $N = 2^d$ was computed, where d is the degree of the primitive polynomial assigned to dimension j, compared with the maximal resolution $\lfloor d/2 \rfloor$, and then the largest difference for those eight projections was measured. Hence, the set of initial values chosen was the one providing the smallest maximal difference. A random search was performed to find these sets of values, since otherwise the search space would have been too large. The optimal values of $m_{j,k}$ obtained for $40 < j \leq 360$ are available from the authors of the used package [13].

The implementation of Sobol's sequence in [13] is based on the Fortran code of Bratley and Fox available at www.acm.org/calgo. The procedure described in [13] could be similarly followed up to dimension j=1000, and was provided by C.Lemieux for the algorithm we are using in our numerical experiments.

Chapter 3

ANOVA Decompostion and Sensitivity Indices

3.1 Analysis of Variance (ANOVA)

The analysis of variance (ANOVA) is a tool devised to describe the dependence of a square-integrable function f on each of the input variables or subgroups of variables. The idea is to decompose f, defined on $[0,1]^s$, as a sum of 2^s components based on each possible subset of variables. This decomposition is very useful to understand the behavior of quasi-Monte Carlo methods for numerical integration methods, where it can be applied to various notions of the effective dimension of an integrand.

Let $f \in L_2$ be a function defined on $[0,1]^s$ and let $\mathbb{A} = \{1, 2, ..., s\}$. For any subset $I \subseteq \mathbb{A}$, let d denote its cardinality, and let $-I$ denote its complementary set in $\mathbb{A}$. A generic point of $[0,1]^s$ is written as $u = (u_1, ..., u_s)$ and u_I denotes the d-vector of components u_i for $i \in I$. We write $[0,1]^d$ for the domain of u_I, and $[0,1]^{s-d}$ for the domain of u_{-I}. The integral over $u_I \in [0,1]^d$ of a function $g(u)$, is a real valued function that depends on u only through u_{-I}.

In the ANOVA decomposition each square integrable function $f(u_1, u_2, ..., u_s)$ is written as a sum

$$f(u) = \sum_{I \subseteq \{1,...,s\}} f_I(u), \tag{3.1}$$

where for any nonempty subset I, $f_I(u)$ depends on u only through u_I. The ANOVA terms are defined by:

$$f_I(u) = \int_{[0,1]^{s-d}} f(u) du_{-I} - \sum_{J \subset I} f_J(u) \tag{3.2}$$

and the ANOVA component $f_\phi(u)$ is simply the integral:

$$I(f) = \int_{[0,1]^s} f(u)du$$

For the decomposition in (3.2) one can show by induction on d that the integral of every summand over any of its own variables is zero:

$$\int_0^1 f_I(u)du_i = 0, \text{ for any } i \in I \tag{3.3}$$

Then it follows that distinct ANOVA terms belong to orthogonal spaces, i.e. if $I \neq J$:

$$\int_{[0,1]^s} f_I \; f_J du = 0,$$

since at least one of the indices in I is not repeated in J and the integral with respect to that variable vanishes (3.3).

Since $f(u) \in L_2$, it can be shown that $f_I \in L_2$, hence the variance of f and f_I:

$$\sigma^2 = \int_{[0,1)^s} f^2(u)du - f_\phi^2$$
$$\sigma_I^2 = \int_{[0,1)^d} f_I^2(u_I)du_I$$

are finite. From the orthogonality of ANOVA terms, it can be easily shown that the variance of f can be be decomposed in a similar fashion as (3.1):

$$\sigma^2 = \sum_{I \subseteq \{1,\ldots,s\}} \sigma_I^2 \tag{3.4}$$

where $\sigma_\phi^2 = 0$. We use σ_I^2 to measure the importance of f_I. Normalized versions σ_I^2/σ^2 are called global sensitivity indices in Sobol' (2001) as we will see in the next section.

3.2 Sobol' Sensitivity indices

In 2001, Sobol' developed a method for global sensitivity analysis of model output, which was named : Sobol' sensitivity indices [28]. The indices were developed based on his earlier work on the Fourier Haar series (Sobol'(1969)). The purpose of the so-called Sobol' sensitivity indices, which we will denote by S_I, is to give us an idea of the relative importance of the term f_I by computing the contribution of its variance to the total variance as follows:

$$S_I = \frac{\sigma_I^2}{\sigma^2} \; \in [0,1]. \tag{3.5}$$

In other words S_I allows us to estimate the sensitivity of a function $f(U)$ with respect to different variables or groups of variables. For instance, $f_I = 0$ if and only if S_I=0 and $f(u)$ is independent of u_i if and only if $S_I = 0$, for all I containing i. S_i, refering to $S_{\{i\}}$ for short, is the main effect of the variable u_i on the output variation. $S_{i,j}$ is the interaction effect, i.e. the fraction of the output variance caused by the two variables together, u_i and u_j, and which cannot be explained by summing the effects of each variable alone. $S_{\{1,\ldots,s\}}$ is the part of the output variation caused by the interaction of the variables altogether and which cannot be explained by summing terms of lower order. Finally, note that $\sum_{I\subseteq\{1,\ldots,s\}} S_I = 1$ using (3.4) and (3.5).

In practice, it is usually neither possible to obtain closed-form expressions for the ANOVA components f_I, nor to compute their variance contributions σ_I^2. Hence, one often needs to estimate Sobol' sensitivity indices. In his work on sensitivity measures for nonlinear models, Sobol recommends using quasi-Monte Carlo methods to approximate these quantities. Informally speaking, if we know, or can guess, which components f_I are important, then we can say that quasi-Monte Carlo integration based on point sets with corresponding high quality projections $P_N(I)$ should give accurate approximations.
For example, the linear MBS problem discussed in chapter 5, is shown to have important one-dimensional components $f_{\{i\}}$, and hence is accurately evaluated using Sobol' sequences due to their excellent one-dimensional projections.

3.3 Computation of Sobol' Sensitivity Indices

As mentioned in the previous section, Sobol' indices can be estimated using quasi-Monte Carlo integration methods. In our numerical experiments, we generate N points, $\{u_1,u_2,\ldots,u_N\}$, where each one is sampled from the Sobol' sequence described earlier in (2.2.1). Then, we compute QMC estimates of the grand mean and total variance in a straighforward way as follows:

$$\hat{f}_\phi = \frac{1}{N}\sum_{i=1}^{N} f(u_i);$$

$$\hat{\sigma}^2 = \frac{1}{N}\sum_{i=1}^{N} f^2(u_i) - \hat{f}_\phi^2$$

It's worth mentioning that it's not necessary to generate a new QMC sample for each group of variables whose sensitivity index is to estimated. For example, for $I = \{j\}$, $\sigma_{\{j\}}^2$, which will be refered to as σ_j^2:

$$\sigma_j^2 = \int_0^1 \left(\int_{[0,1]^{s-1}} f(u)du_{-j} \right)^2 du_j - \hat{f}_\phi^2$$

can be estimated using the following QMC estimator:

$$\hat{\sigma}_j^2 = \frac{1}{N}\sum_{i=1}^{N} f(u_{i,j}^{(1)}, \mathbf{u}_{i,-j}^{(1)}) f(u_{i,j}^{(1)}, \mathbf{u}_{i,-j}^{(2)}) - \hat{f}_\phi^2$$

where the superscripts (1) and (2) refer to two different QMC samples, namely Sobol' in our experiments. To obtain the different samples (1) and (2), we generate a $2s$-dimensional Sobol' point set and use the first s coordinates to define the first sample and the last s ones to define the second sample. That is:

$$\mathbf{u}_i^{(1)} = (u_{i,1}, u_{i,2}, ..., u_{i,s}) \quad \text{and} \quad \mathbf{u}_i^{(2)} = (u_{i,s+1}, u_{i,s+2}, ..., u_{i,2s})$$

for $i = 1, 2, ..., N$. Hence, to compute $\hat{S}_i$, the estimate of the sensitivy index S_i, we need to multiply the values of $f(u)$ obtained at the first sample of N points by the corresponding values of f obtained at the second sample of N points except for the variable u_i, whose data is kept from the first sample. For $\hat{S}_i$ we resample all variables but u_i, where as for $\hat{S}_j$, we resample everything except u_j. Then, after subtracting $\hat{f}_\phi^2$ from the average of the N obtained products, we divide the result by $\hat{\sigma}^2$. For subsets I containing more than one index, Sobol' and Levitan (1999) suggest looking at the following quantity:

$$\gamma_I = \frac{1}{\sigma^2}\sum_{\phi \neq J \subseteq I} \sigma_J^2 \tag{3.6}$$

where QMC methods can be used to estimate this quantity by:

$$\hat{\gamma_I} = \frac{1}{\hat{\sigma}^2}\left(\frac{1}{N}\sum_{i=1}^{N} f(u_i^{(1)}) f(u_I^{(1)}, u_{-I}^{(2)}) - \hat{f}_\phi^2\right) \tag{3.7}$$

using the fact that:

$$\gamma_I = \frac{1}{\sigma^2}\int \left(\int f(u) du_{-I}\right)^2 du_I$$

where, as defined before for the Sobol' indices, $u_I^{(1)}$ represents coordinates $i \in I$ taken from the first sample, and $u_{-I}^{(2)}$ represents coordinates $i \notin I$ taken from the second sample.

The procedure can be summarized as follows:

- Assuming that all S_i need to be computed, we generate two sets of N points, each sampled from a Sobol' sequence as described earlier. Save each set in a matrix of dimension $N \times s$, where s is the dimension of the integrand. One of these matrices is used for sampling and the other one for what we call resampling.

- To compute $\hat{S}_i$, the estimator of S_i, multiply the values of $f(u)$ obtained at the first sample of N points by the corresponding values of f obtained at the second sample of N points excluding the variable u_i, substract $\hat{f}_\phi^2$ from the average of the N products obtained, and then divide the result by $\hat{\sigma}^2$.

- Higher order interactions or sensitivity indices of groups of variables can be obtained in a similar fashion. $\hat{S}_I$ the estimator of S_I, can be obtained by induction on (3.6) and (3.7) as follows:

$$\hat{S}_I = \frac{1}{N}\sum_{i=1}^{N} f(u_i^{(1)}) f(u_I^{(1)}, u_{-I}^{(2)}) - \sum_{J \subset I} \hat{S}_J - \hat{f}_\phi^2$$

  where $I = \{i_1, ..., i_d\} \subseteq \{1, ..., s\}$ and $-I = \{1, ..., s\} \backslash I$

hint : If variable u_i is a relatively important variable, then high values of the first term in the $(f \times f)$ product are multiplied by corresponding high values of the second term, leading to a high $\hat{\sigma}_i^2$ and hence $\hat{S}_i^2$. Otherwise, high values of one of the terms will be multiplied by low ones and a lower σ_i^2 is obtained. This approach was suggested in Saltelli et al. (1997) and even earlier described in a Russian article [27].

3.3.1 Bootstrap Confidence Intervals

To have an idea of how accurate our estimated Sobol' sensitivity indices are, we need to estimate their sampling variability. In Sobol'(1999) the 'probable error' for the variances σ_i^2 was used to measure the accuracy i.e. $\Delta\sigma_i^2$ is calculated so that $Pr\{|\sigma_i^2 - \hat{\sigma_i^2}| \leq \Delta\sigma_i^2\} = 0.5$. However, a better estimate of accuracy for S_i indices can be obtained using Bootstrap Confidence Intervals (BCIs). This procedure is usually applied with independent pseudo-random sampling, however it is still valid with quasi-random sampling as shown in Appendix A of [2].

The idea behind BCIs is to draw, with replacement, N values of the QMC sampled values $\{u_1, u_2, ..., u_N\}$ and repeat this procedure B times until we have generated B samples $\{u_1^b, u_2^b, ..., u_N^b\}$, where $b = 1, 2, ..., B$. For each obtained sample, recalculate the estimate S_i^{*b} of the corresponding variable S_i. The computed B values S_i^{*b}, where $b = 1, 2, ..., B$, are used as a bootstrap estimate of the sampling distribution of the sensitivity index S_i.

In this paper, we will use the moment method which relies on large sample theory and gives a symmetric 95% interval for S_i, as follows:

$$\hat{S}_i \pm 1.96 \times e.s.e(\hat{S}_i)$$

where $e.s.e(\hat{S}_i)$ is the estimated standard error of the bootstrap estimates:

$$e.s.e(\hat{S}_i) = \sqrt{\frac{1}{B-1}\sum_{b=1}^{B}(S_i^{*b} - \bar{S}_i^*)^2}$$

and $\bar{S}_i^*$ is their mean:

$$\bar{S}_i^* = \frac{1}{B}\sum_{b=1}^{B} S_i^{*b}$$

3.3.2 Examples

We investigate the effectiveness of the estimation of the Sobol' indices and the bootstrap intervals described earlier in this section using the following test function:

$$f(U) = \prod_{i=1}^{s} g_i(u_i) \quad \text{where} \quad g_i(u_i) = \frac{|4u_i - 2| + a_i}{1 + a_i},\ a_i \geq 0,\ i = 1, ..., s$$

where f is defined on the s-dimensional cube $[0,1]^s$. The integral of f for all values of the parameter $a_i \geq 0$ is equal to 1. This test function was used in Davis & Rabinowitz (1984) to test mutidimensional integration. The plots of the function for different values of the parameter a_i are given on Figure 3.1. Knowing that u_i lies in [0,1] for each $1 \leq i \leq s$, the function g_i is bounded by:

$$1 - \frac{1}{1 + a_i} \leq g_i \leq 1 + \frac{1}{1 + a_i} \tag{3.8}$$

That is the range of g_i depends on the parameter a_i for all $1 \leq i \leq s$. Hence, the smaller the parameter a_i is, $i = 1, 2, ..., s$, the bigger is the range of g_i and the value of $f(U)$ respectively. For example:

$a_i = 0 \Rightarrow 0 \leq g_i \leq 2$; hence u_i causes a variation in $g \Rightarrow u_i$ is an important variable,
$a_i = 9 \Rightarrow 0.9 \leq g_i \leq 1.1$; u_i causes less variation in $g \Rightarrow u_i$ is less of an important variable,
$a_i = 99 \Rightarrow 0.99 \leq g_i \leq 1.01$; u_i causes little variation in $g \Rightarrow u_i$ is not an important variable.

The advantage of using this test function is the existence of analytic solutions for its sensitivity indices. It's strongly nonlinear and nonmonotonic which provides a good test of the performance of Sobol' methods. As shown in [20], we have that:

$$\sigma_I^2 = \prod_{j \in I} \frac{1}{3(1 + a_j)^2} \tag{3.9}$$

$$\Rightarrow \sigma^2 = \sum_{I \subseteq \{1,2,\ldots,s\}} \prod_{j \in I} \frac{1}{3(1+a_j)^2} \tag{3.10}$$

For our experiments, we choose the dimension $s = 20$ variables, where the parameters $a_i = ((i-1)/2)$, i=1,...,s. Hence, the importance of u_i decreases as i increases.

The Sobol' sensitivity indices were estimated using Sobol' sequences with $N = 2^{12}$ as described in (3.3). Table (3.1) contains a list of ranking of the variables, the exact Sobol' indices S_i, the estimated sensitivity indices $\hat{S}_i$, the moment BCI computed as described earlier, the absolute errors of estimated indices using Sobol' sequence, then the last column lists the absolute errors of the estimated indices using pseudo-random sequences.

From the results, we notice that there is a remarkable monotonic decrease in magnitude, that is in importance, for all the twenty variables as expected. In Sobol' [2], using a smaller N =512, some of the estimates obtained for the higher variables (variable 19 and 20) were negative. This is theoretically impossible, since the indices are a ratio of variances. However, this occurs due to the fact that we are approximating these variances, which implies the result could be underestimating and hence could become negative. But, this still does not affect the message of the results with respect to the relative importance of the input variable.

It seems from the BCI results that the sampling distribution of the estimated Sobol' indices is symmetrical. This hypothesis is supported by the fact that the estimates are the exact midpoints of the BCI's. Notice how $\hat{S}_2$ is lower than the endpoint of the BCI for variable u_1, which makes u_1 significantly more important than any other variable.

We conclude from this experiment that the estimated Sobol' sensitivity indices clearly point out the order of importance of the input variables. The BCI intervals proved to be effective in the assessment of the overlap between these variables, however it is worth mentioning that confidence intervals is not a formal test of the hypothesis that one variable is more important than another, but only to assess how probable it is the case. The bootstrap intervals could also be used to examine the bias in sensitivity analysis estimate, and correct this bias if necessary, as described in [2].

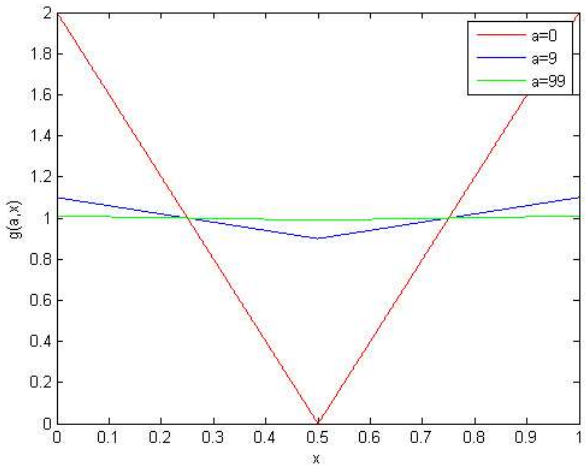

Figure 3.1: : Some examples of the test function.

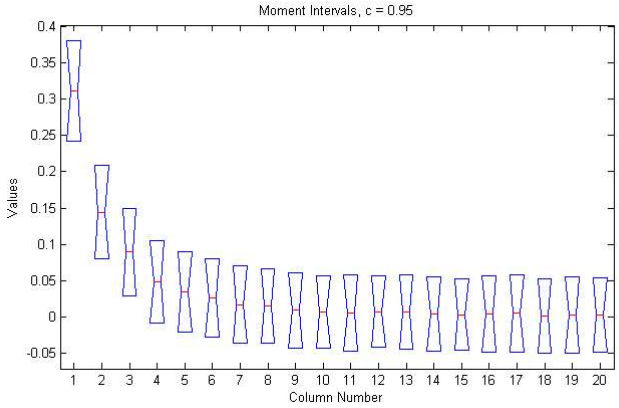

Figure 3.2: : Moment bootstrap intervals for the estimated Sobol' Sensitivity Indices

Table 3.1: : Estimation of Sobol' indices and Bootstrap Confidence Intervals

Variable	S_i	$\hat{S}_i$	***Moment BCI***	**Sobol *e***	**Random *e***
1	0.3065	0.3109	(0.2417, 0.3800)	0.0044	0.0142
2	0.1362	0.1444	(0.0835 , 0.2053)	0.0082	0.0113
3	0.0766	0.0893	(0.0250 , 0.1536)	0.0127	0.0157
4	0.0490	0.0480	(-0.0086, 0.1045)	0.0010	0.0301
5	0.0340	0.0347	(-0.0190, 0.0884)	0.0007	0.0305
6	0.0250	0.0265	(-0.0265, 0.0794)	0.0015	0.0305
7	0.0191	0.0170	(-0.0384, 0.0724)	0.0021	0.0388
8	0.0152	0.0155	(-0.0377, 0.0688)	0.0003	0.0475
9	0.0122	0.0093	(-0.0433, 0.0620)	0.0029	0.0351
10	0.0101	0.0072	(-0.0439, 0.0583)	0.0029	0.0294
11	0.0086	0.0058	(-0.0443, 0.0559)	0.0028	0.0295
12	0.0073	0.0072	(-0.0438, 0.0582)	0.0001	0.0360
13	0.0063	0.0073	(-0.0421, 0.0567)	0.0010	0.0365
14	0.0054	0.0042	(-0.0462, 0.0545)	0.0012	0.0321
15	0.0048	0.0030	(-0.0485, 0.0544)	0.0018	0.0294
16	0.0042	0.0042	(-0.0464, 0.0548)	0.0000	0.0327
17	0.0038	0.0052	(-0.0463, 0.0567)	0.0014	0.0383
18	0.0034	0.0017	(-0.0509, 0.0543)	0.0017	0.0314
19	0.0030	0.0028	(-0.0476, 0.0532)	0.0002	0.0335
20	0.0028	0.0021	(-0.0496, 0.0537)	0.0007	0.0297

Chapter 4

The Effective Dimension

4.1 Definition

The concept of effective dimension is a means for assessing the difficulty of a multidimensional integration problem by studying its ANOVA decomposition. The effective dimension is used to determine how many variables u_i of $f(u)$ are relatively important, in other words, contribute to a considerable variation of $f(u)$. It was introduced to explain the effectiveness of quasi-Monte Carlo integration methods on problems in finance with nominal high dimensions. As mentioned earlier, the error bounds of quasi-Monte Carlo integration is of size $O((logN)^k N^{-1})$ or even $O((logN)^k N^{-3/2})$, which suggests that an improvement in convergence rates over MC methods is not attainable at feasible sample sizes. However, many problems of high dimensions have reported a substantial improvement from using quasi-Monte Carlo at relevant sample sizes. This leads us to assume that even if the integrand has a high nominal dimension, it may be of lower effective dimension. For instance, the mortgage-backed security problem studied in Caflisch, Morokoff, and Owen (1997) has been accurately computed using QMC in dimensions as high as 360, which suggests that the integrand in this problem may be a sum of lower dimensional integrands i.e. lower effective dimension. It is not surprising then that the good low dimensional equidistribution properties of low discrepancy sequences give QMC methods good performance on these problems at relevant sample sizes.

Definition The effective dimension of f in the superposition sense (and in proportion p) is the smallest integer d_S such that :

$$\frac{1}{\sigma^2} \sum_{I:|I|\leq d_S} \sigma_I^2 \geq p$$

The effective dimension of f in the truncation sense (and in proportion p) is the smallest integer d_T such that

$$\frac{1}{\sigma^2} \sum_{J:J\subseteq\{1,2,...,d_T\}} \sigma_J^2 \geq p$$

Hence, a function of effective dimension d in the superposition sense can be well approximated by a sum of functions of at most d variables each. A function of an effective dimension d in the truncation sense can be well approximated by a sum of functions involving the first d variables only.

As mentioned earlier for Sobol' indices, the effective dimension quantities usually cannot be evaluated analytically. The effective dimension in the superposition sense, d_S, can be approximated by using QMC to compute $\hat{\sigma}_I^2$ increasing $|I|$ until:

$$\sum_{I:|I|\leq \hat{d}_S} \hat{\sigma}_I^2 \geq p\ \hat{\sigma}^2 \tag{4.1}$$

Hence, the estimated effective dimension $\hat{d}_S$ is the smallest $|I|$ obtained for which (4.1) holds. For estimating the effective dimension in the truncation sense d_T, we can use the quantity $\hat{\gamma}_I$ defined by Sobol' and Levitan (1999) (see Section 3.3). We compute $\hat{d}_T$ by calculating $\hat{\gamma}_I$ for subsets $I = \{1, 2, ..., d\}$ increasing d until $\hat{\gamma}_I \geq\ p$. Hence, $\hat{d}_T$ is the smallest value of d obtained for which $\hat{\gamma}_I \geq p$ holds.

Below we present two main types of integrands and estimate their effective dimension in the superposition and truncation sense, with proportion p=0.99. A very useful description of the test functions used in comparative studies involving QMC can be found in Owen (2003). We will present a simple linear function and describe its characteristics, and a multiplicative function will follow respectively.

4.1.1 Examples

1. Consider a **linear function** of the form:

$$f(u) = f_0 + \sum_{j=1}^{s} c_j(u_j - 1/2),\ c_j \in \Re, \tag{4.2}$$

which is already decomposed into its ANOVA components:

$$f_j(u) = c_j(u_j - 1/2),\ j = 1, ..., s.$$

where $f_I(u) = 0$ for all subsets I containing more than one index. We can analytically compute σ_j^2 and σ^2 as follows:

$$\begin{aligned} E(f_j) &= \int_0^1 c_j(u_j - 1/2)\, du_j = c_j(1/2 - 1/2) = 0 \\ \sigma_j^2 &= \int_0^1 f_j^2\, du_j - (E(f_j))^2 \\ &= \frac{c_j^2}{3}[(\frac{1}{2})^3 - (\frac{-1}{2})^3] - 0 \\ &= \frac{c_j^2}{3}\frac{1}{4} \\ \sigma_j^2 &= \frac{c_j^2}{12} \\ \Rightarrow \sigma^2 &= \sum_{j=1}^{s} \frac{c_j^2}{12} \end{aligned}$$

Hence, we can compute the exact global sensitivity indices as follows:

$$S_j(u) = \frac{{c_j}^2}{\sum_{j=1}^{s} c_j^2} \; for\; j = 1, ..., s. \tag{4.3}$$

Let's consider two linear functions of the same type as above but with different set of parameters c_j :

$$\begin{aligned} (i)\ s = \ 20,\ c_j = c, \quad c \in \Re, \quad \forall\, j \\ (ii)\ s = \ 40,\ c_j = c^j, \quad c \in (0,1), \quad \text{for } j = 1, ..., s \end{aligned}$$

From (4.3), we have that $\sum_{j=1}^{s} S_j(u) = 1$ for any constants c_j, thus it follows that the effective dimension, in the superposition sense, of any linear function (4.2), must be 1. However, the effective dimension in the truncation sense does depend on these constants c_j. For instance, for the function (i), where all c_j are equal, $d_T = \lceil 0.99s \rceil$. Whereas for function (ii), d_T is the smallest integer d such that:

$$\sum_{I \subseteq \{1,2,..,d\}} \sigma_I^2 = \frac{c^2(1 - c^{2d})}{1 - c^2} \geq 0.99\sigma^2 = 0.99\frac{c^2(1 - c^{2s})}{1 - c^2}$$

After rearranging, we get the exact value of d_T as follows:

$$d_T = \left\lceil \frac{\log(1 - 0.99(1 - c^{2s}))}{2\log c} \right\rceil \tag{4.4}$$

We conducted numerical experiments to test the methods described above for estimating the effective dimension in both superposition and truncation sense. Applying quasi-Monte Carlo integration, we were able to verify that the effective dimension in the superposition sense for both functions (i) and (ii) is equal to 1, using a Sobol' sequence of a quite feasible size (N=4096). Using the same sequence, we verified that $d_T = \hat{d}_T = 20$ for (i) and obtained the results listed Table 4.1 for $\hat{d}_T$ of a function of the same form as (ii), for different values of c and s. Notice that the results obtained for $\hat{d}_T$ in Table 4.1 are exactly the same as d_T given by (4.4).

Table 4.1: : Estimated Effective dimension in the truncation sense for linear function $f(u) = f_0 + \sum_{j=1}^{s} c^j(u_j - 1/2)$, for different values of c and s

$c \backslash s$	**5**	**10**	**20**	**50**	**100**
0.99	5	10	20	50	97
0.95	5	10	20	41	45
0.9	5	10	18	22	22
0.5	4	4	4	4	4
0.2	2	2	2	2	2

2. Now consider the **multiplicative function** studied previously in Section 3.3

$$f(u) = \prod_{j=1}^{s} \frac{|4u_j - 2| + a_j}{a_j + 1}$$

Using (3.9) and (3.10), it can be proved that the effective dimension in the superposition sense of function f is equal to 3. This means that the function $f(u)$ can be well approximated by a sum of functions depending on no more than 3 variables each. We were able to verify this result using the algorithm developed in our numerical experiments and the same Sobol' sequence used previously for estimating the effective dimension of the linear function in experiment 1 above.

Now let's consider different choices of a_j. As we can see from (3.8), the smaller a_j is, the more important the variable u_j is. For example, for $a_1 = a_2 = 0, a_3 = ... = a_8 = 3$ in dimension 8, we found that d_S= 3 and d_T= 8 which is the same result we would get analytically for this case.

Also several values for a_j were used in Wang and Hickernell (2000) for example:

(i) $a_j = 0.01$,
(ii) $a_j = 1$,

(iii) $a_j = j$,
(iv) $a_j = j^2$,
for $j = 1, 2, ..., s$.

Theoretically, as we go from (i) to (iv), the effective dimension (in the truncation sense) d_T of the function f should decrease since a_j increases. Using the same Sobol' sequence as in the above experiments, we were able to obtain the following values for the estimated effective dimension in the truncation sense for the four cases above, where $s = 50$:

(i) $d_T = 42$
(ii) $d_T = 41$
(iii) $d_T = 23$
(iv) $d_T = 4$.

In order to better assess the overall quality of the Sobol' method, we also studied the choice $a_j = (s-j+1)^2$ for $1 \leq j \leq s$, which can be seen as the same choice as (iv), but where we reverse the order of the coordinates of each point. Based on the previously used Sobol' sequence, we found that $d_T = 48$ which means the effective dimension d_T becomes larger than the previous choices since now the most important variables are the last ones u_s, u_{s-1}, and so on.

4.2 Reducing The Effective Dimension

The fact that many high dimensional problems in finance have a low effective dimension in the superposition sense or the truncation sense has no universally accepted explanation, but Sloan and Woźniakowski [26] may offer a possible answer to the outperformance of Quasi-Monte Carlo over Monte Carlo for such problems. What is interesting for us though is that having a low effective dimension in the superposition sense implies that we can obtain accurate results by using a highly uniform quasi-random sequence with good low dimensional projections, i.e. points whose first few coordinates are more uniformly distributed than the latter ones. However, in many constructions of quasi-random sequences such as Sobol' and Halton, the uniformity of low dimensional projections $P_N(I)$ deteriorates with the increase in the coordinates of the point set. Thus, in order to obtain accurate results using such sequences, it becomes necessary for the integrand to have a low effective dimension in the truncation sense as well. The question that naturally arises is: *Is there a technique to reduce the effective dimension in the truncation sense of a financial problem?*

The Brownian bridge construction for pricing of financial derivatives is one of those techniques. This technique aims to modify the way we simulate paths of stochastic vari-

ables, so that the resulting integrands depend heavier on the first few coordinates of the point set, which are presumably more uniform. Later in our numerical experiments, we'll apply the Brownian bridge construction to two different financial problems, pricing digital options and Mortgage backed securities, and analyze its performance on both. But let's us first define this technique and discuss its details.

4.2.1 Brownian Bridge Construction

The Brownian Brigde (BB) technique for QMC integration was first suggested by Caflisch and Moskowitz [6] and then generalized by Morokoff and Caflisch [17]. The authors of Ackworth *et al.* [1] state that 'It attempts to use the best coordinates of each point to determine most of the structure of a path'. Being based on this idea for pricing financial derivatives, the BB method deals with the simulation of the Brownian motion paths in particular, without being concerned with the way the asset prices are combined by the payoff function.

Following a similar procedure to that outlined in A. Papageorgiou [22], let $\{U_t, 0 \leq t \leq T\}$ be a Gaussian Markov process, which is sampled at s times $0 \leq t_1 \leq ...t_s \leq T$. This results in a normally distributed random vector $U = (U_{t_1}, ..., U_{t_s})$. We assume that its mean is zero and its covariance matrix is C. Let $f\colon \Re^s \to \Re$ be a given function and let $I(f) = E[f(U)]$ be the integral we want to compute. Then U can be simulated using $u = (u_1, ..., u_s)$, where u_j are independent normal random variables with mean zero and variance one. For example, assume the underlying asset follows the Black-Scholes model, that is, under the risk-neutral measure (see, e.g., Glasserman (2004)):

$$dS(t) = rS(t)dt + \sigma S(t)dB(t)$$

where $B(.)$ is a standard Brownian motion. Therefore, $S(t)$ has a lognormal distribution, $S(0)$ is the initial price, and σ is the volatility of the underlying asset. More precisely, we can write:

$$S(t) = S(0)e^{(r-\sigma^2/2)t+\sigma B(t)}$$

In the standard approach to generate a path, the coordinates $\{u_1, ..., u_s\}$ of u are successively used to generate the observations $B(t_1), ..., B(t_s)$ of the assets underlying Brownian motion as follows:

$$B_{t_{j+1}} = B_{t_j} + \sqrt{t_{j+1} - t_j}\, u_{j+1},\ j\ =\ 0, ..., s-1,$$

where $B_0 = 0$. Equivalently, in matrix notation we have:

$$\begin{bmatrix} B_{t_1} \\ B_{t_2} \\ \vdots \\ B_{t_s} \end{bmatrix} = M \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_s \end{bmatrix}$$

where M is obtained from the Cholesky factorization of the matrix $C = \{\min(t_i, t_j)\}_{i,j=1}^{s} = M\ M^T$. When $\Delta t = T/s = t_{j+1} - t_j, j = 0, ..., s-1$, i.e. the t_j's are equally spaced which is often the case in practice, the matrix M is given by:

$$M = \sqrt{\Delta t} \begin{bmatrix} 1 & & & \\ 1 & 1 & & \\ \vdots & \vdots & \ddots & \\ 1 & 1 & \dots & 1 \end{bmatrix}$$

The expectation of any integrable function of the discretized path of the Brownian motion is given by :

$$E[f(B_{t_1}, ..., B_{t_s})] = E[f(Mu)],\ u\ = (u_1, u_2, ..., u_s)^T.$$

If instead one tries to use the first few coordinates of u to decide as much as possible the behavior of $B(.)$, then hopefully, this should reduce the effective dimension of the problem in the truncation sense. The Brownian bridge construction does this by first generating B_T, then using this value, and $B_0 = 0$, it generates $B_{T/2}$. It generates $B_{T/4}$ using B_0 and $B_{T/2}$, and it generates $B_{3T/4}$ using $B_{T/2}$ and B_T. The construction proceeds recursively generating the discretely sampled Brownian path by filling in the mid points of the subintervals $T, T/2, T/4, 3T/4, ..., (s-1)T/s$ according to:

$$\begin{aligned} B_T &= \sqrt{T} u_1 \\ B_{T/2} &= \frac{1}{2} B_T + \frac{\sqrt{T}}{2} u_2 \\ B_{T/4} &= \frac{1}{2} B_{T/2} + \frac{\sqrt{2T}}{4} u_3 \\ B_{3T/4} &= \frac{1}{2}(B_{T/2} + B_T) + \frac{\sqrt{2T}}{4} u_4 \\ &\vdots \\ B_{(s-1)T/s} &= \frac{1}{2}(B_{(s-2)T/s} + B_T) + \frac{\sqrt{T}}{2s} u_s \end{aligned}$$

This results in a matrix M different from that of the Cholesky factorization, where $MM^T = C$. The Brownian bridge can be generalized to include unequal length intervals. For $t_{j+1} = t_j + \Delta t,\ j\ =\ 0, ..., s-1, \Delta t\ = T/s$, we can simulate a future value $B_{t_k}, k > j$, (given the value B_{t_j}) according to :

$$B_{t_k} = B_{t_j} + \sqrt{(k-j)\Delta t}\ u$$

where u follows a normal distribution $N(0,1)$. We can also simulate B_{t_i} at any intermediate point $t_j < t_i < t_k$ (given the values of B_{t_j} and B_{t_k}) according to the Brownian bridge formula :

$$B_{t_i} = (1-\gamma)B_{t_j} + \gamma B_{t_k} + \sqrt{\gamma(1-\gamma)(k-j)\Delta t}\; u,$$

where u follows a normal distribution $N(0,1)$ and $\gamma = (i-j)/(k-j)$

Although the Brownian bridge (or, another similar construction) can provide better convergence in a number of interesting cases, we shall see that this advantage is not guaranteed because there exist cases for which it does not perform well. One of these cases is the digital option, where the underlying asset is lognormally distributed. For this option, the dimension by truncation obtained using the Brownian bridge is no less than that obtained using the standard discretization (i.e., the one reflecting the Cholesky decomposition of the covariance matrix of the Gaussian distribution), even when the dimension is as small as 2. In [22], A. Papageorgiou claims that we can conclude the following :

- The Brownian bridge does not offer a consistent advantage in QMC integration
- We still need a clarification for the argument attributing its success to its ability to reduce the effective dimension of a problem
- A covariance matrix decomposition can be interpreted as a change to the integrand or to the sample points which may yield a harder problem.

Returning to the integral $I(f)$, the Monte Carlo error is not affected by the choice of the covariance matrix decomposition because this error depends on the first and second moments of f which remain invariant under the different decompositions. In general, this is the case for any method with an error depending only on the moments of f. On the other hand, the choice of the matrix M, $MM^T = C$, does affect the quasi- Monte Carlo error. This is due to the fact that any choice M can be interpreted as a change in the integrand or as a change in the sample points, and both are factors upon which the QMC deterministic error bound depends.

Chapter 5

Problems from Finance

5.1 Randomized Quasi-Monte Carlo For Obtaining Error Estimates

In order to assess the performace of QMC, we need to obtain bounds on its integration error. Bounds on the integration error using QMC can be obtained and are valid for functions satisfying strong regularity conditions (e.g., f must be of bounded variation: see Niederreiter (1992) and Owen (2005) for the details). However, these bounds are very difficult to compute and too conservative and hence not useful in practice. An alternative way to assess the peformance of QMC is described below using what we call a Randomized Quasi-Monte Carlo sequence.

One way of obtaining error estimates for QMC methods is to randomize the underlying quasi-random sequence. Using the same argument as C. Lemieux [12], let v be a uniform random vector in some space Ω. Then choose a randomization function r such that $r\colon \Omega \times [0,1)^s \to [0,1)^s$ and construct the randomized version $\widetilde{P_N} = \{\tilde{u_1}, ..., \tilde{u_N}\}$ of P_N, defined by $\tilde{u_i} = r(v, u_i)$. For example, with the Cranley-Patterson rotation (Cranley and Patterson, 1976), $\Omega = [0,1)^s$ and $r(v, u_i) = (u_i + v) \bmod 1$.

The function r should be chosen so that:
(i) $r(v, u)$ is uniformly distributed over $[0,1)^s$ for any u.
(ii) $\widetilde{P_N}$ has the same highly uniform properties as P_N.

Once a randomization is chosen, the variance of the resulting estimator $\sum_{i=1}^{N} f(\tilde{u_i})/N$ can be estimated by generating m i.i.d. randomized point sets $\widetilde{P_N}$. For more on randomization techniques and standard constructions for QMC methods, we refer the reader to Owen (1998), L'Ecuyer and Lemieux (2002), and Glasserman (2004).

We discussed in chapter 4 the concept of Effective Dimension and Brownian bridge technique used to enhance it. In the following sections we are going to use the methods proposed by Sobol' as discussed earlier, to estimate the effective dimension of two problems: a digital option and a mortgage backed security. Using randomized Sobol' sequences, we will investigate the accuracy of the QMC approximations for these two problems for different dimensions, and compare our results to those of MC obtained using pseudo-random sequences. We will also check the effect of applying the standard Brownian bridge technique on their effective dimension in the truncation sense .

For both problems discussed below, QMC integration methods are performed using Sobol' sequences generated by the same package described in Section 2.2.1. To obtain the RQMC results, Sobol' sequences are randomized by a random shift. That is, a randomized point set $\widetilde{P_N}$ is obtained by shifting all the points in P_N by a constant value randomly drawn from the uniform distribution $U([0,1]^s)$. Thus, properites (i) and (ii) discussed earlier remain valid.

5.2 Digital Option

We chose this problem because it has been shown (Papageorgiou (2002)) that the Brownian bridge technique does not reduce the effective dimension by truncation. The payoff of a digital call option is given by :

$$C_D = \frac{1}{s}\sum_{j=1}^{s}(S_{t_j} - S_{t_{j-1}})_+^0 S_{t_j}$$

where $t_j = jT/s$ for j=1,...,s, and $(x)_+^0$ is equal to 1 if $x > 0$, and is 0 otherwise. Thus the value of this type of option is determined more heavily by local trends of the underlying asset rather than by its global trend, which might be a reason for the failure of the Brownian bridge technique on this problem.

Table 5.1 gives results of applying RQMC for a digital option with $T = 1$, $r = 0.045$, σ= 0.3, and $S(0) = 100$. The number of randomizations m was set to 25 for these results. For each pair (s, N), we give the estimator for the price of the option on the first line, and its standard error on the second line. As we can see in this table, the RQMC method consistently succeed in reducing the variance for this problem.

Applying the Brownian bridge technique to the digital option, we obtained larger values for the effective dimension in the truncation sense than that of the standard discretization. The failure of the Brownian bridge verifies our claim that the price of this option depends on the local changes in the price of the underlying asset more heavily than the global change. Table 5.2 below shows the values of D_T for different dimensions s.

Table 5.1: : Digital Option

Variable **(s,N)**	**MC**	**Sobol'**
(64,1024)	52.6191	52.6926
	0.1552	0.01
(64,4096)	52.6101	52.7007
	0.07	0.0014
(128,1024)	52.2402	52.26226
	0.0879	0.0068
(128,4096)	52.2061	52.2222
	0.0414	0.0012
(256,1024)	51.904	51.9377
	0.0936	0.0039
(256,4096)	51.7997	51.9105
	0.0276	0.0013

Table 5.2: : Effective Dimension in truncation sense of the Digital Option

s	**without BB**	**with BB**
64	62	64
128	121	128
256	236	255

5.3 Mortgage Backed Security

A Mortgage-Backed Security (MBS) is an asset-backed security or debt obligation that represents a claim on the cashflows generated by a group of one or more mortgages. Since they started in the 1970s, mortgage-backed securities have undergone significant growth and acquired a huge popularity as an investment tool among individuals and financial institutions. This was favored by the benefits offered to investors by MBSs, including attractive yields, minimal credit risk and a liquid market. In 2002, the total value of outstanding MBS was already exceeding $3.8 trillion and constituted 21% of the total outstanding bond debt in the U.S. Fixed Income market (Kelman (2002)).

The valuation process of mortgage-cashflows-dependent securities is based on modeling the borrowers' refinancing strategy. In the early models suggested by Brennan and Schwartz (1985), the borrower was thought of as an optimal agent actively seeking to minimize the present value of his mortgage. This model was inefficient because it failed to regard borrowers as suboptimal agents and thus match the actual prepayment rates. This resulted in developing behavioral mortgage-valuation models that produced reduced form prepayment behaviour models.

5.3.1 Details

A Mortgage-Backed Security is created when a mortgage issuer assembles together a set of mortgages into pools. MBSs are issued by government agencies, government-sponsored enterprises, and private entities. Most MBSs are issued by three government sponsored agencies: Gennie Mae (GNMA), Freddie Mac (FHLMC) and Fannie Mae (FNMA). Mortgages are assembled into pools based on specific government guidelines and common properties, such as interest rate, payment terms, etc. When the pool is set up, the issuer has the right to sell units to investors directly or through securities markets. Ownership of the MBS entitles the investor to receive collections of the mortgage borrowers interest and principal payments minus the expenses of the deal such as the issuers fees for guaranteeing the timely payments to the investor. The payments on the MBS are guaranteed by the respective issuing agency. For more details, see "MBS basics" by Mark Adelson, Nomura Securities International, Inc. ($http://www.securitization.net/pdf/Nomura/MBSBasics_31Mar06.pdf$)

Thus, the MBS owner faces two different risks: the interest and prepayment risks. Fluctuating interest rates influence the interest paid by the borrower and fluctuating prepayment rates influence the principal on which the interest is paid. Because of all these factors, the valuation of MBSs becomes a complex process that involves modelling the interest rates and the prepayment behaviour of the individuals, which in turn depends on term structure, strategical and other random factors.

5.3.2 Term Structure and Prepayment Models

As mentioned in the previous section, the value of an MBS is essentially sensitive to fluctuations in the interest rate, which makes its model a very important factor in the MBS valuation. The Rendelman-Barter model is the simplest way to model the interest rate r, where r follows the geometric Brownian motion model:

$$dr = \mu(t, r)rdt + \sigma(t, r)rdB$$

where μ is the drift term, σ is the volatility and B is a Brownian motion.

Note that this model assumes the same behaviour for a short-term interest rate as that of a stock price. This assumption, however, is deficient since unlike stocks, interest rates drift toward a long-run mean which is called a mean reversion property. Although the Rendelman-Barter model does not exhibit this property, we need to set drift term $\mu = 0$ to eliminate any price drift. A more sophisticated model with the mean reversion property, like the Vasicek model, could also be used.

The prepayment model is another crucial factor in MBS valuation. Building this model entails many difficult tasks, since it aims to describe the different behaviours of individuals when it comes to paying off their mortgages. One of these difficult tasks is to accurately describe rational behaviour in response to changing financial factors, for example refinancing due to falling interest rates.

Many prepayment models are suggested in the literature. But most models agree that the following four factors should be taken into consideration: (1) prevailing mortgage rate, (2) characteristics of the underlying mortgage pool, (3) seasonal factors, and (4) general economic activity. Details of these factors can be found in "Valuation of fixed income securities and derivatives" By Frank J. Fabozzi [8].

5.3.3 Model Details

In this section, we will introduce the same model and notation used by Paskov (1997), and Caflisch, Morokoff, and Owen (1997). The model discussed below assumes that the cash flows consist of interest and principal repayment, and therefore depend on the future interest and prepayment rates. Let our MBS be of length M=360 months with fixed initial interest rate i_0 i.e. the current interest rate at the begining of the mortgage. Let $\{\xi\}_{k=1}^{M}$ be normally distributed random variables with mean 0 and variance σ^2.

The interest rate at month k, i_k, is given by:

$$i_k = K_0 e^{\xi_k} i_{k-1} = K_0^k i_0 e^{\xi_1 + \ldots + \xi_k}$$

where the constant $K_0 = e^{-\sigma^2/2}$ is chosen to normalize the log-normal distribution so that $E(i_k) = i_0$. The initial interest rate i_0 is an additional constant that needs to be specified. The prepayment model, which is a function of the above interest rates, is:

$$\begin{aligned} w_k &= K_1 + K_2 \arctan(K_3 i_k + K_4) \\ &= K_1 + K_2 \arctan(K_3 K_0^k i_0 e^{(\xi_1 + \ldots + \xi_k)} + K_4) \end{aligned}$$

where K_1, K_2, K_3, K_4 are constants of the model. w_k represents the fraction of outstanding mortgages at month k that are prepaying in that month. Using this model for the prepayment rate, we can obtain several sample problems by modifying the parameter values $i_0, K_1, K_2, K_3, K_4, \sigma$, hence modifying the prepayment behaviour in response to changing interest rates. K_1 and K_2 are respectively the mean and volatility of the prepayment rate, while K_3 and K_4 together control the linearity of the function arctan. For example, small magnitudes of K_3 and K_4 make arctan linear and large values make it non-linear, with the sign of K_3 controlling the sign of the slope in the linear case.

The total cash flow from r_k remaining mortgages for month k consists of two parts: $r_k w_k$ of the mortgages are prepaying the entire mortgage, thus paying Cc_k (where C is the monthly payment and c_k is the remaining mortgage annuity after month k), while the remaining $r_k(1 - w_k)$ are paying only the monthly payment C. Thus m_k, the cash flow for month k, is the following:

$$\begin{aligned} r_k &= \prod_{j=1}^{k-1} (1 - w_j) \\ m_k &= C r_k ((1 - w_k) + w_k c_k) \\ c_k &= \sum_{j=0}^{360-k} (1 + i_0)^{-j} \end{aligned}$$

The present value of a Mortgage-Backed Security is the discounted value of all underlying cash flows in the security.

$$PV = E(\sum_{k=1}^{360} u_k m_k)$$

where E is the expectation over the random variables involved in the interest rate fluctations and u_k is the discounting factor for month k, $u_k = \prod_{j=0}^{k-1} \frac{1}{1+i_j}$

Numerical examples in this paper use two problems defined by two sets of parameters provided by Caflisch, Morokoff, and Owen (1997).

The first set results in what was refered to as a Nearly Linear problem, defined by:

$$(i_0, K_1, K_2, K_3, K_4, \sigma^2) = (0.007, 0.01, -0.005, 10, 0.5, 0.0004)$$

The second set results in a problem refered to as Nonlinear, defined by:

$$(i_0, K_1, K_2, K_3, K_4, \sigma^2) = (0.007, 0.04, 0.0222, -1500, 7.0, 0.0004)$$

We are interested in these two problems since they belong to two different classes. In Caflisch, Morokoff, and Owen (1997), the first problem was shown to be very nearly a linear function of the random variables ξ_k. Using Latin hypercube sampling, they found that 99.96% of the variation comes from one-dimensional structure, that is has an effective dimension of one in the superposition sense. In our experiments, we used a Sobol' sequence of size $N = 2^{18}$ (starting from the point 2^{10} according to the algorithm described in Section 2.2.1). We found that the Nearly Linear example appears to be a nearly linear function of u_k since (99.73%) of the function variation is explained by the variation of the first 76 variables :

$$\sum_{i=1}^{76} S_i \geq 0.99$$

Hence most of the variation of the funcion comes from the one dimensional structure. On the other hand, we were not able to obtain such results for the effective dimension of the nonlinear problem in the superposition sense due to the sensitivity of the prepayment function to extremely low interest rates as shown in figures (5.1) and (5.2).

Table 5.3 gives results of applying RQMC for both the linear and nonlinear MBS problems described above. The number of randomizations m was also set to 25 for these results. For each dimension s, we give the estimator for the value of the MBS on the first line, and its standard error on the second line. As we can see in this table, the RQMC method consistently succeed in reducing the variance for this problem.

Applying the Brownian bridge technique to the MBS, we were able to obtain a much smaller effective dimension in the truncation sense than that obtained from the standard discretization. Table 5.4 below shows the values of D_T for both the linear and nonlinear problems..

Table 5.3: : MBS

N	**MC**	**Sobol'**
nearly-linear		
1024	131.6667	131.5931
	0.047	1.90E-04
4096	131.5694	131.5895
	0.0119	4.12E-05
non-linear		
1024	130.5558	130.5609
	0.0059	1.30E-03
4096	130.5654	130.553
	0.0098	5.60E-04

Table 5.4: : Effective Dimension in truncation sense of MBS

problem	**BB**	**without BB**
linear	34	144
nonlinear	34	63

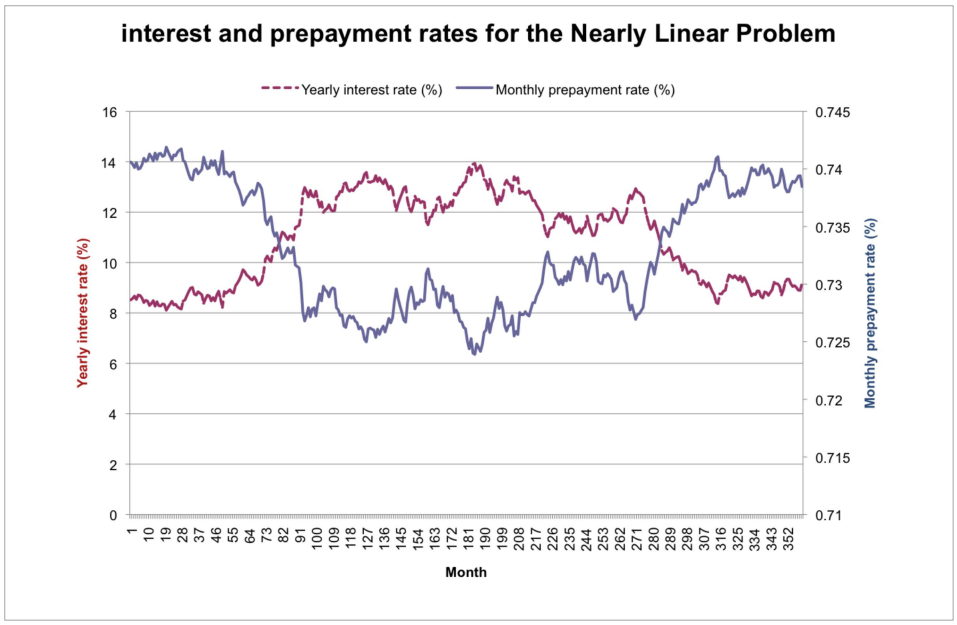

Figure 5.1: : Sample Interest and Prepayment Rates for the Nearly Linear Problem

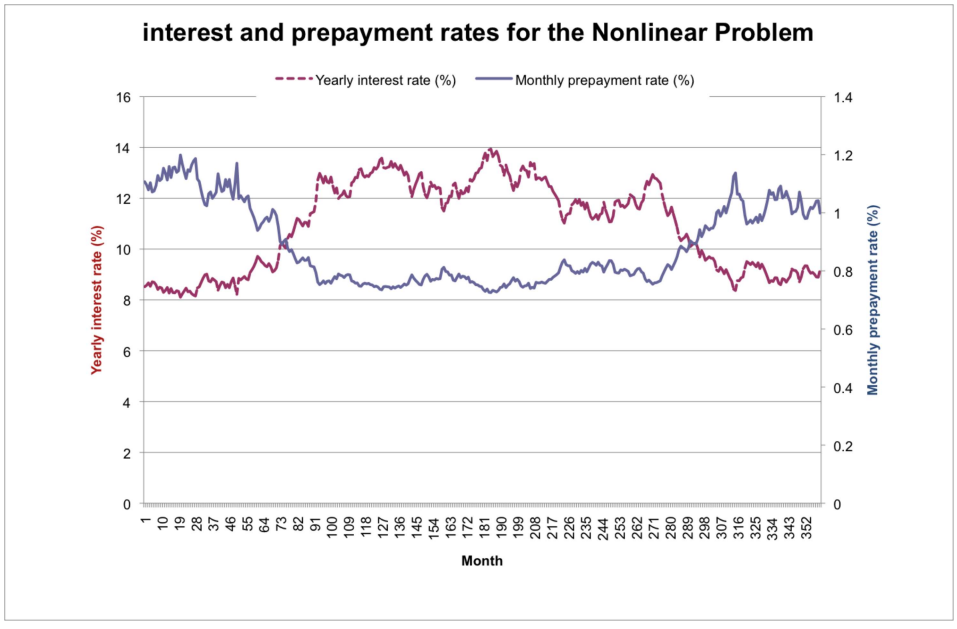

Figure 5.2: : Sample Interest and Prepayment Rates for the Nonlinear Problem

Chapter 6

Conclusion

Many finance problems use small time increments i.e. large number of input variables which leads to their formulation as high-dimensional integrands. However, it was found that these problems are essentially low-dimensional, in the sense that the majority of their variation is governed by their low order ANOVA components, usually first and second order. This property together with good inherent features of the quasi-random sequence renders QMC integration method more efficient for approximating these integrands regardless of the fact that they are high dimensional.

To investigate the efficieny of QMC method and the notion of effective dimension attributed to it, we presented in this paper a powerful method for sensitivity analysis given by Sobol'. This method incorporates advantages from several existing approaches, from fast fourier transforms to correlation ratios. It adds to the previous methods the ability to carry out a full ANOVA on the outcome of a numerical experiment. Results from the literature and the experiments performed by Sobol' [2] on low-dimensional problems have encouraged us to apply it on high-dimensional finance problems with amenable features. For these problems, we have shown that using quasi-random Sobol' sequences with good low-dimensional projections, the estimated Sobol' sensitivity indices were able to identify which and how many input variables are important, thus giving a good estimate of their effective dimension. For example, for the digital option, we have shown that all its input variables are important regardless of their order, in dimensions as high as 256. As for the MBS problem, we were able to show that (99.73%) of its variation come from its one-dimensional ANOVA components in dimension as high as 360. That is, it could be well approximated by sum of one-dimensional functions. The efficiency of QMC was also revealed in the results obtained from randomized Sobol' sequences for both problems, where we were able to obtain estimators with much smaller variance than MC.

Finally,we have investigated the Brownian bridge technique to reduce the effective dimension of the presented high-dimensional problems. This technique was proved to be

very successful for the MBS problem where more than (99%) of its variation was reduced to the first 34 input variables. However, this was not the case for digital options, meaning that all its input variables are equally important.

In this paper, we have focused on specific finance problems and tried to detect their special features, such as their effective dimension, showing the efficiency of applying QMC on these problems. However, answering the question of why QMC methods work well in high dimensions is a challenging task which has been extensively studied in the recent years. The research focuses on two main ideas. First, the features of the functions that arise in practice, such as the notion of low effective dimension used by Paskov and Traub(1995), Carlisch el al(1997), Paskov(1997), and Wang and Fang(2003), the "weigthed" function spaces presented by Sloan and Woznaikowski(1998), the isoptropic and non-isotropic properties presented by Papageorgiou (2001,2003), etc. Second, the aspect of point sets, where it was shown non-constructively by Heinrich *et al.*(2001) that there exists a point set $\{u_1, u_2, ..., u_N\}$, such that its star dicrepancy satisfies $D^*(\{u_i\}) \leq C^* \sqrt{s/N}$ where C^* is a constant.

APPENDICES

In what follows we will describe in brief the algorithm used to obtain the results of our numerical experiments. First, we have to call the function named *allDriver* to initialize the global variables that are going to be used in every function such as s, N, etc... *allDriver* also calls a function *generateSamples* to generate the two Sobol' samples needed for computing estimates of Sobol' sensitivity indices (SSIs).

Notice from the computation procedure of these SSIs (Section (3.3)), the computation of higher order SSIs requires the the values of lower order ones. Thus, to avoid computing the same $\hat{S}_I$ twice, we will save the computed $\hat{S}_I$'s in the vector *allSi* where each entry corresponds to a unique subset I. These subsets are saved in another matrix named *allsubs*. For example if the dimension s=3, *allDriver* intitializes *allSi* and *allsubs* as follows:

$$\text{allSi} = \begin{bmatrix} c \\ c \\ c \end{bmatrix} \text{ where allsubs} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix}$$

where c is a constant. Now, let's introduce the major function in our algorithm, namely the function *allSSI*. This function takes I as an input and saves the computed value $\hat{S}_I$ in its proper location in the vector *allSi*. For example, after calling *allDriver*, if we call *allSSI*([1]), that is with {1} as an input, *allSSI* will compute $\hat{S}_{\{1\}}$ and save it in the first entry of the vector *allSi*. If we call *allSSI*([3]), the computed value $\hat{S}_{\{3\}}$ will be saved in the third entry of *allSi*, and so on. Now, if we call *allSSI*([1, 3]) to compute $\hat{S}_{\{1,3\}}$, we don't have to compute $\hat{S}_{\{1\}}$ and $\hat{S}_{\{3\}}$ again, but we just need to extract their values from *allSi*. We also have to expand *allsubs* and *allSi* in order to include the new subset {1,3} and $\hat{S}_{\{1,3\}}$ respectively.

The efficiency of this procedure is more evident when we are estimating the effective dimension in the superposition sense because we will have to compute all $\hat{S}_{\{I\}}$'s for increasing values of $|I|$ until their total sum exceeds the proportion p. The procedure for now is very simple, but what if the input I of *allSSI* has subsets J whose corresponding $\hat{S}_{\{J\}}$'s have not yet been computed?

In our algorithm, we have created the functions *subsets* and *allsubs* in order to deal with this case. The function $subsets(L, I)$ generates all subsets of I of length L, and is used by $allsubs(I)$ in order to generate all the power set of I. So, when $allSSI(I)$ is called, the procedure is summarized as follows:

1. Generate all subsets of I

2. For each subset J, check if the corresponding entry of $allSi$ is still equal to c

   - if the entry is not equal to c, that is $\hat{S}_{\{J\}}$ value has been computed and has replaced c, extract its value from $allSi$

   - if the entry is still equal to c, then call $allSSI(J)$ with J as an input to compute $\hat{S}_{\{J\}}$

This procedure is repeated recursively until the value of $\hat{S}_{\{I\}}$ is computed. This is a brief description of the procedure, in fact, other functions are involved to perform the above steps. for example, when we call $allSSI(K)$ with $|K|$ greater than $|I|$, for all previously input $|I|$'s, we have to expand our vector of subsets *allsubs*. Also, to find the index of the entry in $allSi$ corresponding to a subset I, we need a function called *subsetind*. For details of the implemention of $allSSI$, we refer the reader to check the matlab code below.

.1 Function *allDriver*

```
% Define the globally used variables
global s  % the dimension
global N            % the size of the sample
global B            % the number of bootstrap samples

% the first and second Sobol' samples
global sample
global resample

global fs % N-by-1 vector of the function values at the sample point set
global f0 % the mean of the function values at the sample point set
global D  % the variance of the function values at the sample point set

% all the sensitivity indices computed are saved in allSi
global allSi
global allsubs

% generateSamples: generates two N-by-s samples, in this case the samples generated
% are going to be Sobol' sequences of N points
generateSamples;

s= 20;
N= 4096;

% initialize the first s elements of allSi randomly
allSi= ones(s,1)*5;
% initialize allsubs to [1,2,...,s]
allsubs= (1:s)';

% f returns a vector of the values of f evaluated at each point in sample
fs= f(sample);
f0= mean(fs);
D= mean(fs.^2) - f0^2;
```

.2 Function for generating Sobol' samples

```
function generateSamples

global s
global N
global sample
global resample

% Call i4_sobolAlt once to initialize its variables.
% i4_sobolAlt generates Sobol' samples point by point
seed=0;
i4_sobolAlt (2*s, seed );

% in this case we will start our Sobol' sequence from the 2^10 point
seed = 2^10;

for i=1:N

     [ bigpoint(i,:), seed ] = i4_sobolAlt (2*s, seed );
end

sample= bigpoint(:,1:s);
resample= bigpoint(:,s+1:2*s);
bigpoint=[];
```

.3 Function for computing SSI's and their BCI's

```
% Input : A subset of {1,2,...,s}
% Output: Si, estimate of Sobol Sensitivity indices of the input subset

function [Si] = allSSI(indices)

global s
global sample
global resample

global allSi
global allsubs
global f0
global D
global fs

% eliminate any possible zero entries in the input indices
ind= indices(logical(indices));
L= length(ind);

% resample all variables except those given by indices
x=resample;
x(:,ind)= sample(:,ind);
fx= f(x);

product= fs.*fx;
Di = mean(product) - f0^2;
Si= Di/D;

% define B the number of bootstrap samples
B= 1000;

% if the input vector has one element, save the above computed Si and compute its BCIs
if L==1
allSi(ind)=Si;
pick_Nsamples_Btimes = ceil(rand(N,B)*N);
Di= mean(product(pick_Nsamples_Btimes)) - f0^2;
Si_B= Di/D;
```

```
mean_Si_B = mean(Si_B);  diff= Si_B-mean_Si_B;
ese = 1.96*sqrt(1/(B-1)*diff*diff');
% lSi is the lower bound and uSi is the upper one
lSi= Si-ese;  uSi = Si+ese;

return
else

% if the input vector has more than one index,
% First expand the vectors allSi and allsubs in case the input vector
% requires the computation of new SSI's other than the ones saved in allSi
[p,q] = size(allsubs);
for j=q+1:L
% newsubsets is a matrix of all new subsets of {1,2,...,s} containing j elements
newsubsets= subsets(j,(1:s));
ln= size(newsubsets,1);
allsubs= [allsubs,sparse(p,1); newsubsets];
allSi= [allSi; ones(ln,1)*5];
end

% Generate all subsets of the input vector
indices_subs= allsubsets(ind);
l= size(indices_subs,1);

% loop over all the subsets and check if the corresponding SSI is already computed
% and saved in allSi or needs to be recursively computed
for i=1:l-1

    subs = indices_subs(i,:);
    k= subsetind(subs);

    % this condition implies that SSI of this subset is computed earlier
    if allSi(k) ~= 5
        subt= allSi(k);
        % Make sure the SSI subtracted is positive otherwise consider it zero
        if subt>0
        Si= Si- subt;
        end

    % otherwise, call the function again to compute the SSI of this subset
```

```
    else
        subt= allSSI(subs);
        if subt>0
        Si= Si- subt;
        end
    end

% k returns the location of ind in the vector allsubs
k= subsetind(ind);
% save the computed SSI in the corresponding location
allSi(k)=Si;
end end
```

.4 Function for computing the effective dimension in the superposition sense

```
function ds= sdim(prop)

global s
global sample

ds = 0;
SUM = 0;
SET= (1:s);

while (SUM<prop) & (ds<1)

    ds = ds+1;
    S = subsets(ds,SET);
    l= size(S,1);

    for i=1:l
        s(i)= allSSI(S(i,:));

        if s>0
        SUM= SUM + s;
        end
    end
```

```
end
```

.5 Function for computing the effective dimension in the truncation sense

```
 function dt= tdim(prop)

global s
global sample
global resample
global fs
global f0
global D
dt = 0 ;
SUM =0;
S=[];

while  SUM<prop dt< s

        dt= dt+1;

% resample all variables except those given by indices to compute their SSI
x= resample;
x(:,(1:dt))= sample(:,(1:dt));
product= fs.*f(x);
Di = mean(product)- f0^2;
SUM= Di/D

end
```

Bibliography

[1] P. Acworth, M. Broadie, and P. Glasserman. A comparison of some Monte Carlo and quasi-Monte Carlo techniques for option pricing. In P. Hellekalek and H. Niederreiter, editors, *Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing*, volume 127 of *Lecture Notes in Statistics*, pages 1–18. Springer-Verlag, New York, 1997. 24

[2] G. E. B. Archer, A. Saltelli, and I. M. Sobol'. Sensitivity measures, ANOVA-like techniques and the use of bootstrap. *Journal of Statistical Computation and Simulation*, 58:99–120, 1997. 14, 16, 36

[3] R. Bellman. *Adaptive Control Processes: A Guided Tour*. Princeton University Press, Princeton, NJ, 1961. 4

[4] PAUL BRATLEY and BENNETT L. FOX. Algorithm 659: Implementing sobol's quasirandom sequence generator. *ACM Transactions on Mathematical Software (TOMS)*, 14:88 – 100, 1988. 8, 9

[5] R. E. Caflisch, W. Morokoff, and A. B. Owen. Valuation of mortgage-backed securities using Brownian bridges to reduce effective dimension. *The Journal of Computational Finance*, 1(1):27–46, 1997. 6

[6] R. E. Caflisch and B. Moskowitz. Modified Monte Carlo methods using quasi-random sequences. In H. Niederreiter and P. J.-S. Shiue, editors, *Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing*, volume 106 of *Lecture Notes in Statistics*, pages 1–16. Springer-Verlag, New York, 1995. 24

[7] P. Davis and P. Rabinowitz. *Methods of Numerical Integration*. Academic Press, New York, second edition, 1984.

[8] Frank J. Fabozzi. *Valuation of Fixed Income Securities and Derivatives*. John Wiley & Sons, 2001. 31

[9] P. Glasserman. *Monte Carlo Methods in Financial Engineering*, volume 53 of *Application of Mathematics – Stochastic Modelling and Applied Probability*. Springer, New York, 2004.

[10] F.J. Hickernell. Quadrature error bounds and figures of merit for quadi-random points. Technical Report MATH- 111, Department of Mathematics, Hong Kong Baptist University, 1996. 6

[11] P. L'Ecuyer and C. Lemieux. Recent advances in randomized quasi-Monte Carlo methods. In M. Dror, P. L'Ecuyer, and F. Szidarovszki, editors, *Modeling Uncertainty: An Examination of Stochastic Theory, Methods, and Applications*, pages 419–474. Kluwer Academic Publishers, Boston, 2002.

[12] C. Lemieux. Randomized quasi-Monte Carlo methods: A tool for improving the efficiency of simulations in finance. In R. G. Ingalls, M. D. Rossetti, J. S. Smith, and B. A Peters, editors, *Proceedings of the 2004 Winter Simulation Conference*, pages 1565–1573. IEEE Press, 2004. 27

[13] C. Lemieux, M. Cieslak, and K. Luttmer. RandQMC user's guide: A package for randomized quasi-Monte Carlo methods in C. Technical Report 2002-712-15, Department of Computer Science, University of Calgary, 2002. 8, 9

[14] Christiane Lemieux. *Monte Carlo and Quasi-monte Carlo Sampling*. Springer Verlag, 2009. 4

[15] W. J. Morokoff and R. E. Caflisch. Quasi-random sequences and their discrepancies. *SIAM Journal on Scientific Computing*, 15:1251–1279, 1994. 7

[16] W. J. Morokoff and R. E. Caflisch. Quasi-monte carlo integration. *J Comp. Phys.*, 122:218–230, 1995. 7

[17] W. J. Morokoff and R. E. Caflisch. Quasi-Monte Carlo simulation of random walks in finance. In H. Niederreiter, P. Hellekalek, G. Larcher, and P. Zinterhof, editors, *Monte Carlo and Quasi-Monte Carlo Methods 1996*, volume 127 of *Lecture Notes in Statistics*, pages 340–352. Springer-Verlag, New York, 1997. 24

[18] H. Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods. SIAM Philadelphia*, 1992. 6, 7

[19] A. B. Owen. Latin supercube sampling for very high-dimensional simulations. *ACM Transactions on Modeling and Computer Simulation*, 8(1):71–102, 1998.

[20] A. B. Owen. The dimension distribution and quadrature test functions. *Statistica Sinica*, 13:1–17, 2003. 15

[21] A. B. Owen. Multidimensional variation for quasi-Monte Carlo. In Jianqing Fan and Gang Li, editors, *International Conference on Statistics in honour of Professor Kai-Tai Fang's 65th birthday*, pages 49–74. World Scientific Publications, Hackensack, NJ, 2005.

[22] A. Papageorgiou. The Brownian bridge does not offer a consistent advantage in quasi-Monte Carlo integration. *Journal of Complexity*, 18(1):171–186, 2002. 24, 26

[23] A. Papageorgiou and J. Traub. Beating Monte Carlo. *Risk*, 9:63–65, June 1996.

[24] S. Paskov and J. Traub. Faster valuation of financial derivatives. *Journal of Portfolio Management*, 22:113–120, 1995. 1

[25] S.P. Paskov. New methodologies for valuing derivatives. In S. Pliska and M. Dempster, editors, *Mathematics of Derivative Securities*, pages 545–582. Cambridge University Press, UK, Isaac Newton Institute, Cambridge, 1997.

[26] I. H. Sloan and H. Woźniakowski. When are quasi-Monte Carlo algorithms efficient for high dimensional integrals? *Journal of Complexity*, 14:1–33, 1998. 23

[27] I. M. Sobol'. *Multidimensional Quadrature Formulas and Haar Functions.* Nauka, Moskow, 1969. In Russian. 14

[28] I. M. Sobol'. Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates. *Mathematics and Computers in Simulation*, 55:271–280, 2001. 11

[29] I. M. Sobol' and Y. L. Levitan. On the use of variance reducing multipliers in Monte Carlo computations of a global sensitivity index. *Computer Physics Communications*, 117:52–61, 1999.

[30] X. Wang and F. J. Hickernell. Randomized Halton sequences. *Mathematical and Computer Modelling*, 32:887–899, 2000.

[31] X. Wang and I. H. Sloan. Why are high-dimensional finance problems of low effective dimension? *SIAM Journal on Scientific Computing*, 27:159–183, 2005.