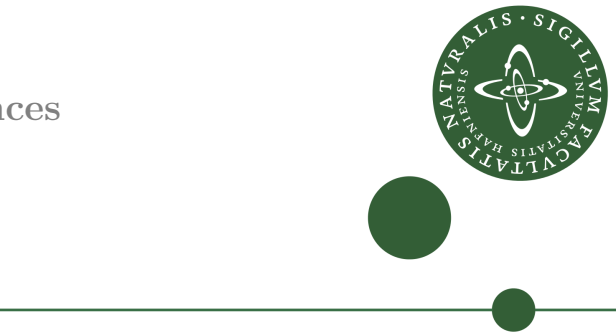
Department of Mathematical Sciences
University of Copenhagen



Quasi-Monte Carlo Method

in The Heston Model

Raza Ali Mahmood

Advisor: David Glavind Skovmann

The 3rd May 2022

Master Thesis in Mathematics - Economics

Department of Mathematical Science
University of Copenhagen

Abstract

Quasi-Monte Carlo methods are permutations on the standard Monte Carlo method, which employs supremely uniform quasirandom numbers rather than Monte Carlo's pseudorandom numbers. This thesis investigates the application of quasi-Monte Carlo methods on the Heston model. Our main focus in this paper is the Broadie-Kaya scheme, which our main algorithms are based on. The Monte Carlo methods provides statistical error estimates; however, this is lost in the quasi-Monte Carlo, but in return provides faster convergence than a standard Monte Carlo. A recent discovery has shown that the randomized quasi-Monte Carlo can preserve the speed of quasi-Monte Carlo, but also reintroduces the error estimates from Monte Carlo methods. For our investigation, we compare the Euler discretization with Full Truncation, the Broadie-Kaya scheme using pseudorandom sequences, then amp up the speed using quasirandom sequences and finally a randomized quasi-Monte Carlo.

Contents

1 Introduction

The modern international stock market was created when the Dutch East India Company was listed as the worlds first publicly traded company in Amsterdam, back in the 1600s. Since then, the field of stock exchange and the common interest in it has increased profusely. We currently have a broad spectrum of investment one is able to make in the financial market without directly buying a stock; such financial derivatives include options, futures, swaps and forwards. All of these types of financial instruments are based on mutually agreed upon contracts between two entities. Usually listed price will be dependent on an underlying asset, such as a stock, index or commodity.

And since the stock market creation, people have been speculating on how one is able to foresee whether an investment in a given derivative will generate profitable returns. After the publication of the Black-Scholes (BS) model in 1973, many mathematical finance researchers have invested large chunks of their time in specifically pricing financial derivatives using this model. Reason being that a European option had a closed-form solution in this model, which made it an attractive from a practical point of view. However, the BS model has various assumptions, such as constant interest rates, no change in volatility and no external shocks. Thus, the model fell short in the financial crashes experienced since 1973, and gradually the introduction of more complex derivates forced financial analysts to develop new models or improve them.

20 years later after its introduction, the arrival of the Heston model constituted a breakthrough that was able to remedy some of the aforementioned shortcomings of the BS model. The Heston model incorporates the stochastic volatility into the option pricing model, while by adding two stochastic processes for stock price and volatility, consequently lead to an increase in model complexity compared to the BS model. It furthermore permits a semi-closed solution for the European Option. Thus to price these European options we will rely on Monte Carlo methods.

The Monte Carlo method, remains a popular tool used to assess pricing of financial

derivates. It has seen large growth in popularity in the last decades, resulting in many specialized applications implementing a Monte Carlo algorithm to solve problems in computational finance.

A variety of problems in computational finance relies on computing the expected values of a random variable defined on some probability space. The Monte Carlo method takes the underlying asset and simulates it using random numbers to estimate the expectation of the random variable by averaging the samples. In this paper, we particularly study the quasi-Monte Carlo (QMC) method, which is rather similar to the standard Monte Carlo method. The fundamental difference between QMC and Monte Carlo is that QMC utilizes the so-called low discrepancy sequence in its simulations.

A low discrepancy sequence, also named quasirandom sequence, is a deterministic sequence (why QMC is ordained the deterministic Monte Carlo method), as opposed to the pseudorandom sequence used in the Monte Carlo method. In this thesis we implement the Sobol sequence, which is a low discrepancy sequence also used by Baldeaux and Roberts, 2012. This choice of sequence is based on the characteristics of the Sobol sequence, which are preferred and we will see why later on. This leads to one of two advantages of the QMC method: it provides a deterministic convergence, as opposed to the stochastic convergence in the Monte Carlo method. Moreover the QMC method estimates convergence to the true solution faster than the standard Monte Carlo

$$\text{QMC: } \mathcal{O}\left(\frac{\log(N)^d}{N}\right), \qquad \text{Monte Carlo: } \mathcal{O}\left(\frac{1}{\sqrt{N}}\right).$$

Where N is the size of the sample and d is the dimension of the problem. Even so, there are some deficiencies that need to be addressed with this method. Firstly, in higher dimensions we observe the low discrepancy sequences degenerate. The error bound given by classical Koksma-Hlawka inequality will be shown in different setting, but that upper bound will hard to compute in practise.

2 Heston Simulation Schemes

As we wish to apply Monte Carlo and QMC on the Heston model, it would be beneficial to explain the dynamics of the Heston model. We recall a filtered probability space $(\Omega, \mathscr{F}, (\mathscr{F}_t)_{t\geq 0}, \mathbf{P})$, where we assume $\mathbf{P}$ is a risk-neutral pricing measure. Thus, the correlated Heston model with a Cholesky decomposition is given by the following stochastic differential equations (SDE[1])

$$dS_t = rS_t dt + \sqrt{V_t} S_t \left[\rho dW_{1,t} + \sqrt{1-\rho^2} dW_{2,t} \right], \tag{2.1}$$

$$dV_t = \kappa(\theta - V_t)dt + \sigma\sqrt{V_t} dW_{1,t}. \tag{2.2}$$

Where W_1 and W_2 represents two independent Brownian-motion processes under the risk neutral measure, $\mathbf{P}$ and with the instantaneous correlation ρ between the two processes $\mathbb{E}[dW_1, dW_2] = \rho d_t$. The Process, S, models the dynamic of the asset and V is the stochastic variance of the asset. Through out this paper, we will be using the following notation $S_t = S(t)$, $V_t = V(t)$, $W_{1,t} = W_1(t)$ and etcetera, which also applies to the formulas above. Here, t is the index for time, r is the risk-free rate of interest, θ is the long-term average variance, κ is the mean rate of reversion and σ is the volatility of volatility, which determines the variance of V_t.

As the Heston model has seen high employment over many decades now, its characteristic function is known for the log-asset price in a closed-form. It was also been shown by Carr and Madan, 1999, that the calibration of vanilla call option in the Heston model can be done efficiently with a fast Fourier transform (FFT).

We will use the characteristic function derived in Broadie and Kaya, 2006, which was obtained through the Laplace transform of $\int_s^t V_u du$ given V_s and V_t. A more detailed

[1]Stochastic Differential Equation

derivation can be found in their appendix. Thus we obtain the characteristic function with the form

$$\Phi(a) = \mathbb{E}\left[\exp\left(ia\int_s^t V_u du \middle| V_s, V_t\right)\right] = \frac{\gamma(a)e^{-\frac{1}{2}(\gamma(a)-\kappa)(t-u)}(1-e^{-\kappa(t-u)})}{\kappa(1-e^{-\gamma(a)(t-u)})}\frac{A(a)}{B}e^{C(a)}. \quad (2.3)$$

Where $A(a)$, B, $C(a)$ and $\gamma(a)$ are defined as such,

$$\begin{aligned}
A(a) :=& I_{\frac{\psi}{2}-1}\left(\sqrt{V_u V_t}\frac{4\gamma(a)e^{\frac{1}{2}\gamma(a)(t-u)}}{\sigma^2(1-e^{-\gamma(a)(t-u)})}\right),\\
B :=& I_{\frac{\psi}{2}-1}\left(\sqrt{V_u V_t}\frac{4\kappa e^{\frac{1}{2}\kappa(t-u)}}{\sigma^2(1-e^{-\kappa(t-u)})}\right),\\
C(a) :=& \frac{V_u+V_t}{\sigma^2}\left(\frac{\kappa(1+e^{\kappa(t-u)})}{(1-e^{-\kappa(t-u)})} - \frac{\gamma(a)(1+e^{-\gamma(a)(t-u)})}{(1-e^{\gamma(a)(t-u)})}\right),\\
\gamma(a) :=& \sqrt{\kappa^2 - 2\sigma^2 ia}.
\end{aligned}$$

Here, $I_v(x)$ denotes the Bessel function of the first kind, where ψ is described as in (2.6). The characteristic function is needed to calculate the second step in Broadie-Kaya scheme., and adds an even larger computational load to an already time consuming scheme.

Analytical Properties of the Variance Process

There are many known results regarding the square-root variance dynamic in the Heston model. One example of this is, the variance process is guaranteed to always be greater than or equal to 0. Specifically, if we set the Feller condition $2\kappa\theta \leq \sigma$, which implies that the variance process will never reach zero. Remark, that this condition is hardly ever satisfied, when calibrating with real world market data.

The variance process' distribution is also known; conditional on V_s for $s < t$, the variance process is distributed as a constant C_0 multiplied by a non-central chi-squared distribution with ψ degree of freedom and the non-centrality parameter λ, i.e.

$$\mathbb{P}(V_t \leq x | V_s) = F_{\mathcal{X}^2_\psi(\lambda)}\left(\frac{x}{C_0}\right), \quad (2.4)$$

here $F_{\mathcal{X}^2_\psi(\lambda)}\left(\frac{x}{C_0}\right)$ describes the cumulative distribution of a non-central chi-squared distri-

bution, specifically

$$F_{\mathcal{X}^2_\psi(\lambda)}(z) = \sum_{i=0}^{\infty} \frac{e^{-\frac{\lambda}{2}} \left(\frac{\lambda}{2}\right)^i}{i!} \frac{\int_0^z z^{\frac{d}{2}} e^{-\frac{y}{2}} dy}{\Gamma\left(i + \frac{k}{2}\right)}, \tag{2.5}$$

where

$$C_0 := \frac{\sigma^2(1 - e^{-\kappa\Delta t})}{4\kappa}, \quad \psi := \frac{4\kappa\theta}{\sigma^2}, \quad \lambda := \frac{4\kappa e^{-\kappa\Delta t}}{\sigma^2(1 - e^{-\kappa\Delta t})} V_s, \text{ with } \Delta t = s - t. \tag{2.6}$$

Although it should be noted that some discretization schemes of the Heston model dynamics rely heavily on these properties, an example of which the Broadie-Kaya scheme, while others, discretization schemes such as Euler or Milstein seen in Lord et al., 2010 relies less on these properties.

Euler Scheme

In the standard Monte Carlo method for option pricing, the technique utilized is to simulate the underlying asset, which is done in discrete time steps. However, the dynamics in Heston model are described by continuous-time stochastic processes, thus a discrete approximation will introduce an bias error. The Euler scheme is the easiest way to apply discretization to our SDEs, we simply approximate the stock process and variance process in a discrete grid. One however needs to be mindful of how negative values are handled in the variance process; the wrong manipulation of the negative values will lead to biased schemes. As seen in Lord et al., 2010, the right procedure leads to a highly efficient Euler scheme, which can outperform many schemes in computational efficiency.

Conditional on time s a Euler discretization of the variation process for $t > s$ with $\Delta t = t - s$ is as follows

$$V_t = V_s + \kappa(\theta - V_s)\Delta t + \sigma\sqrt{V_s} Z_V \sqrt{\Delta t}. \tag{2.7}$$

Here we have that Z_V is standard normal distributed variable for the variance process. There is plenty of literature that delves into the main problem of the scheme in (2.7), on how one is to handle the negative values of the variance if that event should occur. The paper Lord et al., 2010 examinates many possible schemes, and the fix that seems most auspicious is the Full Truncation scheme, which leads to the following discretization of the variance process

$$V_t = V_s + \kappa(\theta - V_t^+)\Delta t + \sigma\sqrt{V_s^+}Z_V\sqrt{\Delta t}, \tag{2.8}$$

where we have used the following notation $x^+ = \max(0, x)$ and similarly for the asset price process the Euler scheme yields

$$S_t = S_s\left(1 + r\Delta t + \sqrt{V_s^+}\left[\rho Z_V + \sqrt{1-\rho^2}Z_S\right]\sqrt{\Delta t}\right). \tag{2.9}$$

As described by Lord et al., 2010, V_s is chosen as a non-negative variable and this Euler scheme does entail a discretization error. Thus, we can alternately use the exact solution to the stock price process seen in equation (2.1), with the application of Ito's lemma is given by

$$S_t = S_s \exp\left(\int_s^t r - \frac{1}{2}V_u du + \int_s^t \sqrt{V_t}S_t\left[\rho dW_{1,t} + \sqrt{1-\rho^2}dW_{2,t}\right]\right). \tag{2.10}$$

Then combine it with log-Euler discretization we obtain

$$\log(S_t) = \log(S_s) + \left[r + \frac{1}{2}V_s^+\right]\Delta t + \sqrt{V_s^+}\left[\rho Z_V + \sqrt{1-\rho^2}Z_S\right]\sqrt{\Delta t}. \tag{2.11}$$

This log-Euler scheme will not introduce any discretization error in the stock price process, but the Euler discretization of the variance process entails some discretization bias. We chose to solve the issue of having a negative V_s in equation (2.11) by only selecting $V_s^+ = \max(0, V_s)$. As seen in Lord et al., 2010 the Ito correction term in equation (2.11) is now consistent with the respective volatility term in the stock price process.

2.2.1 Full Truncation Algorithm

With this, the log-Euler scheme for the stock price process with the Full Truncation scheme for the Heston model can be boiled down to the following steps:

1 Create a random sample from a standard normal distribution, Z_1 and set $Z_V := Z_1$.

2 Condition on V_s, compute V_t from equation (2.8).

3 Create a new random sample from a standard normal distribution, Z_2, and define

$$Z_S = \rho Z_V + \sqrt{1-\rho^2}Z_2.$$

4 Calculate $\log(S_t)$ given $\log(S_s)$ using the equation in (2.11).

Broadie & Kaya's Exact Scheme

In this paper we will be using standard Monte Carlo, quasi-Monte Carlo and Randomized quasi-Monte Carlo method to price any contingent claim. We will utilize the same approach as described in Broadie and Kaya, 2006, where they work out an exact simulation scheme for the Heston Model.

To begin the exact scheme we will discuss the incorporated difficulties in the scheme: by how to generate an exact sample from the distribution of S_t by conditioning on the values generated from the variance process. If we continue on the solution from (2.10), and apply Ito with log price dynamics, $\log(S_t)$,

$$\log(S_t) = \log(S_s) + \frac{1}{2}\int_s^t V_u du + \rho \int_s^t \sqrt{V_u} dW_{2,u} + \sqrt{1-\rho^2}\int_s^t \sqrt{V_u} dW_{1,u}. \tag{2.12}$$

Inserting the square-root variance process from equation (2.2) gives us the following input

$$\begin{aligned} V_t =& V_s + \int_s^t \kappa\left(\theta - V_u\right) du + \sigma \int_s^t \sqrt{V_u} dW_{2,u}. \\ V_t =& V_s + \kappa\theta\Delta t - \kappa \int_s^t V_u du + \sigma \int_s^t \sqrt{V_u} dW_{2,u}. \end{aligned} \tag{2.13}$$

Equivalently be written as

$$\int_s^t \sqrt{V_u} W_{2,u} = \sigma^{-1}(V_t - V_s - \kappa\theta\Delta t + \kappa \int_s^t V_u du). \tag{2.14}$$

Here it is remarked in Broadie and Kaya, 2006, one can incorporate equation 2.14 into the solution of 2.12 this leads to following

$$\begin{aligned} \log(S_t) =& \log(S_s) + \frac{\kappa\rho}{\sigma}\int_s^t V_u du - \frac{1}{2}\int_s^t V_u du \\ & \frac{\rho}{\sigma}(V_t - V_s - \kappa\theta\Delta) + \sqrt{1-\rho^2}\int_s^t \sqrt{V_u} dW_u, \end{aligned} \tag{2.15}$$

As we are interested in generating an exact sample from the distribution of S_t by conditioning on the values generated from the variance process. We must clearly sample the three stochastic quantities seen in equation (2.15). Furthermore the Broadie-Kaya scheme can be written up in a 3-step fashion based on these three stochastic quantities:

1 Compute $V_t|V_s$ from the quantities (2.4) and (2.6); thus one will be able to use

that $V_t|V_s$ has a distribution akin to a constant C_0 times non-central chi-squared distribution with ψ degree of freedom and the non-centrality parameter λ.

2 Now we compute $\int_s^t V_u du|V_s, V_t$ and we use the characteristic function derived from Broadie and Kaya, 2006

$$\tau(a, V_s, V_t) = \mathbb{E}\left[e^{ia\int_s^t V_u du|V_s,V_t}\right] = \Phi(a). \tag{2.16}$$

Thus we can take the above mentioned characteristic function and invert it numerically to gain the value of the distribution function $G(x)$ for a certain point $x \in \Omega$. Essentially,

$$G(x, V_s, V_t) = \frac{2}{\pi}\int_0^\infty \frac{\sin(ax)}{x}\mathrm{Re}\left[\tau(a, V_s, V_t)\right] da$$

Then to produce the sample $\int_s^t V_u du|V_s, V_t$ we use,

$$G\left(\int_s^t V_u du\Big|V_s, V_t\right) = U,$$

and from here we will invert G over a uniform random sample U to obtain x_i: $x_i = G^{-1}(U, V_s, V_t)$, e.g. finding the root of the following $G(\int_s^t V_u du|V_s, V_t) - U = 0$. Using the Newton-Raphson root search involves multiple Fourier inversions for each evaluation of $G(x_i, V_s, V_t)$.

3 Final step is to compute $\int_s^t \sqrt{V_u} dW_u|\int_s^t V_u du$. Since V_u is independent of $W_{2,u}$, it follows directly that this expression can be distributed as $N(0, \int_s^t V_u)$. Therefore sampling for this quantity, we simply sample from a normal distribution.

2.3.1 Broadie & Kaya Algorithm

In our implementation in MATLAB of the Broadie-Kaya scheme algorithm we have been inspired by page 296 in Kienitz and Wetterau, 2013 for our own algorithm. The steps for the exact algorithm to compute the three stochastic quantities in equation (2.12) is feasible with the following 5-step implementation:

1 We generate a sample of V_t by conditioning on V_s and using the definitions from (2.6). The sample is constructed by sampling from a constant multiplied with a non-central chi-squared distribution with ψ degrees of freedom and a non-centrality

parameter λ.

2 Generate the sample of $\int_s^t V_u du$ byconditioning on V_s and V_t, using numerical inversion of the distribution function $G\left(\int_s^t V_u du | V_s, V_t\right)$ over the uniform sample U. This implies that you need to find the root search of the following expression $G\left(x_i, V_s, V_t\right) - U = 0$. Note that the distribution function G does not have a known closed-form, thus we have to obtain it by Fourier inverting the characteristic function of $\int_s^t V_u du | V_s, V_t$.

3 Use the (2.14) to determine:

$$\int_s^t \sqrt{V_u} dW_{2,u} = \sigma^{-1}(V_t - V_s - \kappa\theta\Delta t + \kappa \int_s^t V_u du) \tag{2.17}$$

4 Now generate an independent sample from a standard normal distribution Z_S and make use of the property that $\int_s^t \sqrt{V_u} dW_u$ is normally distributed with mean zero and variance $\int_s^t \sqrt{V_u} du$ and therefore we can write it as

$$\int_s^t \sqrt{V_u} dW_u \sim Z_S \sqrt{\Delta t \int_s^t V_u du}, \tag{2.18}$$

5 Now we have the three quantities $\int_s^t \sqrt{V_u} dW_{2,u}$, $\int_s^t \sqrt{V_u} dW_{1,u}$, $\int_s^t V_u du$ and given $\log(S_s)$ we can calculate equation (2.15), and finally obtain $\log(S_t)$.

Downside of The Exact Scheme

In this section we will be discussing the practical limits when one tries to adopt this method. We will delve into the implementation issues that are integrated when we aim to use the Broadie-Kaya scheme.

Therefore we start at the beginning, the first step in the Broadie-Kaya scheme, which is about how the variance process $V_t | V_s$ is sampled from a constant C_0 times a non-central chi-squared distribution with ψ degrees of freedom and the non-centrality parameter λ.

$$V_t \stackrel{d}{=} C_0 \mathcal{X}^2_\psi(\lambda) \tag{2.19}$$

It is possible to change the representation of the non-central chi-squared distribution to

$$\mathcal{X}^2_\psi(\lambda) \overset{d}{=} \begin{cases} (Z+\sqrt{\lambda})^2 + \mathcal{X}^2_{\psi-1}(\lambda) & \text{for } \psi > 1 \\ \mathcal{X}^2_{\psi+2N}(\lambda) & \text{for } \psi > 0 \end{cases} \tag{2.20}$$

Where we have that $Z \sim N(0,1)$, λ is from (2.6), let $\mathcal{X}^2_v$ be an ordinary chi-squared distribution with v degree of freedom and N is a Poisson distributed variable with mean $\mu = \frac{\lambda}{2}$. In the case of $\psi << 1$ we are generally forced to work with second representation in (2.20)[2]. Therefore we need to condition on the Poisson variate and successively generate a sample from the chi-squared or gamma distribution to obtain the exact sampling from the variance process[3].

As mentioned in Broadie and Kaya, 2006 it is possible to generate gamma variates through the accept and reject method. This method is typically slower when compared to sampling methods on normal variates, relatively speaking. A major drawback in the Broadie Kaya method is specifically the acceptance and rejection technique dependency on the number of samples for the specific Heston model parameters. Incorporating the acceptance and rejection scheme for the simulation of V_t is therefore inconvenient. In Andersen, 2007 is noted that it is a problem in risk management application, as the perturbation of model parameters will alter the total number of pseudorandom numbers required to generate a path of S and V. They also include that if a seed were to be used for pre- and post-perturbation paths, the emerging correlation between sample path payouts pre- and post perturbation stays low, which leads to a noisy estimate[4].

Lastly in this section we will note the second step in the Broadie & Kaya algorithm, where we want to generate a sample of $\int_s^t V_u du | V_s, V_t$ by a numerical inversion of the distribution function $\left(\int_s^t V_u du | V_s, V_t\right)$ on the uniform variable U to do the following root search of $G(\int_s^t V_u du | V_s, V_t) - U = 0$. Since the distribution function of G has no known closed-form, we need to obtain it by Fourier inverting the characteristic function (2.3). Here we use two modified Bessel functions that are representing an infinite series. Dialling it back to the root detecting approach (which requires Fourier inversions) has to be repeated as long as the tolerance level ε has not been satisfied, so for a guess of x_i that allow the

[2]For $d > 1$ we would work with the first representation as it is easier to sample from the normal distribution versus the Poisson distribution.

[3]Chi-squared is a special case of the gamma distribution, $\mathcal{X}^2_v \overset{d}{=} \Gamma(\frac{v}{2}, 2)$, where $\Gamma(k,\theta)$ is a gamma distribution with shape k and scale θ

[4]The effectiveness of the calculation of model sensitivities crucially depends on the size of the correlation coefficient between pre- and post perturbation paths.

following to hold true $G(x_i, V_s, V_t) - U < \varepsilon$.

This stands next to the detail that in both the evaluation of (2.3) and the corresponding Fourier inversions needed, substantial computational endeavour is demanded. As such, this step of the algorithmic implementation requires a great deal of caution to elude from notable biases from the numerical inversion.

3 Standard Monte Carlo

We wil briefly cover a standard Monte Carlo method and summarize some of the known facts discovered through time to compare it with a quasi-Monte Carlo. The problem we are considering is an integral of a Lebesgue integrable function $f(\mathbf{x})$ can be formulated as the expectation of the function $f(\mathbf{x})$.

$$B[f] = \int_{I^d} f(\mathbf{x})dx = \mathbb{E}[f(\mathbf{x})]. \tag{3.1}$$

Here we have that elements of $\mathbf{x}$ are independent uniformly distributed in the unit cube, $I^d = [0,1]^d$. Then if you sample $\{x_n\}$ sampled from a uniform distribution. Consider then the basic Monte Carlo approximation.

$$B_N[f] = \frac{1}{N}\sum_{n=1}^{N} f(x_n). \tag{3.2}$$

Then since we have that x_n is i.i.d with a uniform distribution and we know all moments, we have by the strong law of large numbers that the approximation converges to the true value with probability 1

$$\lim_{N\to\infty} \frac{1}{N}\sum_{n=1}^{N} f(x_n) \to \mathbb{E}[f(\mathbf{x})]. \tag{3.3}$$

Thus resulting in that the estimator being unbiased. From Central Limit Theorem (CLT) we have that, for large enough N

$$f(x_n) - B[f]] \overset{d}{=} \frac{\sigma}{\sqrt{N}}\nu, \tag{3.4}$$

in which $\nu \sim \mathcal{N}(0,1)$ random variable and the constant $\sigma = \sigma[f]$ as the standard deviation of Monte Carlo method, written as follows

$$\sigma[f] = \sqrt{\mathbb{E}\left[\left|\int_{I^d} f(x_n) - B[f]\right|^2\right]}. \tag{3.5}$$

Then the variance of $B_N[f]$ becomes

$$\mathrm{Var}(B_N[f]) = \mathrm{Var}\left(\frac{1}{N}\sum_{n=1}^{N} f(x_n)\right) = \frac{1}{N}\mathrm{Var}\left(f(x_n)\right)$$

Here we used that the variance of sums of independent variables is the sum of the variances. Using the fact that the estimator is unbiased, the root mean error (RMSE) is equal to the standard deviation of $B_N[f]$,

$$\begin{aligned}\mathrm{RMSE} &= \left[\mathbb{E}\left(\left|\frac{1}{N}\sum_{n=1}^{N} f(x_n) - \mathbb{E}[f(\mathbf{x})]\right|^2\right)\right]^{\frac{1}{2}} \\ &= \left[\mathrm{Var}\left(\frac{1}{N}\sum_{n=1}^{N} f(x_n)\right)\right]^{\frac{1}{2}} \\ &= \frac{1}{\sqrt{N}}\mathrm{Var}\left(f(x_n)\right)^{\frac{1}{2}} = \frac{\sigma[f]}{\sqrt{N}}\end{aligned}$$

The RMSE will be an estimator that determines the quality of our estimate for the true value on average. Note here that $\sigma[f]$ does in fact not depend on N, meaning that the order of convergence on the RMSE and the convergence rate of the Monte Carlo method is $\mathcal{O}\left(\frac{1}{\sqrt{N}}\right)$.

As seen in the partial proof using CLT in Caflisch, 1998 it becomes quite apparent that the Monte Carlo integration error is of an order $\mathcal{O}(\frac{1}{\sqrt{N}})$, which is a great result since no matter the dimension of the problem we will have this order on the error. Duly note that this only gives a probabilistic error bound - there is no absolute upper bound on the error in theory. The reason for it being a probabilistic bound and not a true upper bound, meaning

$$\mathbb{P}\left(|f(x_n) - B[f]| < \varepsilon\right) = 1 - \alpha,$$

for some specified level of confidence α. Implying that it is just a probability for the error being lower than $\frac{1}{\sqrt{N}}$, and not an upper bound. Nevertheless, we are pleased with its performance in the numerical comparison.

Another difficulty of the Monte Carlo method for numerical integration stems from the specification on the nodes being independent random samples. The difficulty being randomness, how does one truly generate random samples? We will simply overlook this philosophical query and employ pseudorandom numbers instead of truly random samples.

We will be using the built in function in MATLAB to create our pseudorandom sample.

Exact vs Euler Scheme

In this section we have some intermediate results of the Monte Carlo method using both the so-called exact and Euler scheme. We will start off with comparing each schemes convergence as we see in the plot below

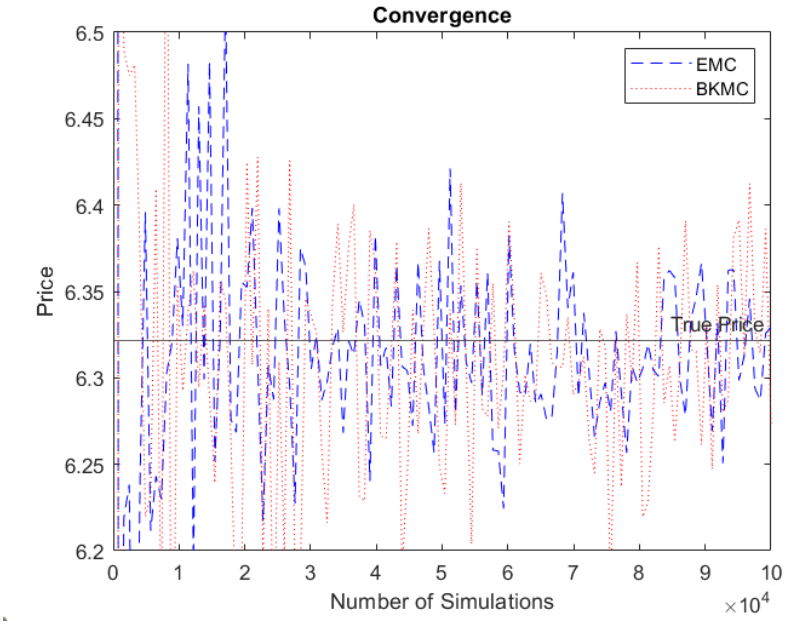


Figure 3.1: The methods EMC (Monte Carlo with Euler scheme with Full Truncation) and BKMC (Monte Carlo with Broadie-Kaya scheme, using pseudorandom sequence) are estimating the European option price, which has the true price, 6.3219 (black line). Option parameters $S = 100$, $K = 100$, $V_0 = 0.04$, $\theta = 0.04$, $\kappa = 0.25$, $\sigma = 0.5$, $\rho = -0.6$, $r = 0$ and $T = 1$. Note that for the Euler scheme we are discretizing with 100 steps a year

In this plot we have taken 124 data points spanning from 1 to 10^5, which is a bit less than Broadie and Kaya, 2006, as more is not really needed to showcase the point of our arguments. From our implementation of the two schemes we see that both prices converges towards the true price of the European option in some fashion. As expected, the oscillation in the price is very high for both schemes in the beginning, but only the EMC method becomes somewhat stable, since for the BKMC we used the pseudorandom number generator in MATLAB called `rand`. It looked like BKMC would stabilise around the 70,000 simulation, but because of randomness the oscillation still persists. In Broadie and Kaya, 2006 they did use a bit north of 10 million in their simulation, which is quite a lot, and in practise it is not really possible as we see from the computational speed in our next plot, it is simply a pipe dream.

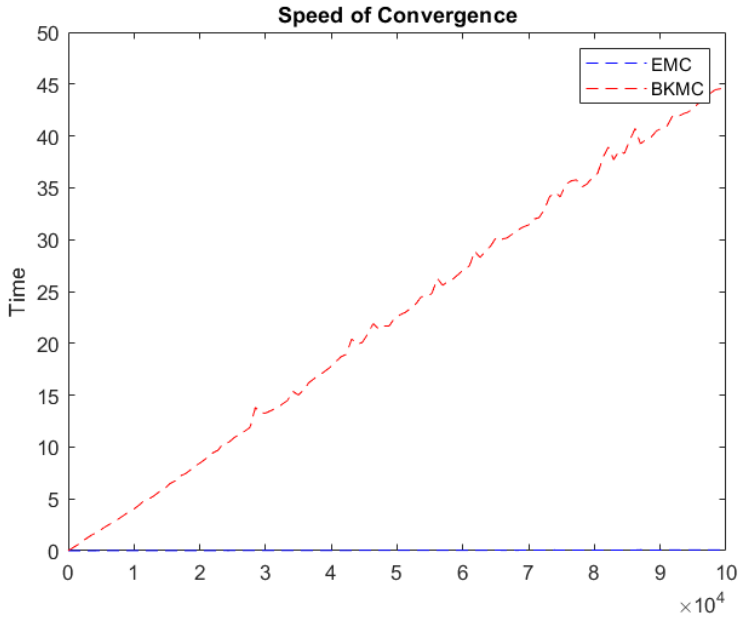


Figure 3.2: Plot of computation time measured in seconds as a function of simulations done. The plot consist of 124 data points of EMC and BKMC time estimates, same option parameters as the previous figure (3.1).

In this plot we are observing BKMC speed being many times slower than EMC, which is to be expected from the rather cumbersome calculations done in BKMC. We already knew this, as it is mentioned in most literature using the BKMC method without quasirandom numbers. The Heston model that we use here is a 3-dimensional problem [1], where we see that the Euler discretization with Full Truncation is simply superior in this dimension. It is worth remarking than in our code we used a stepsize of 100 a year for the Euler discretization, which according to Broadie and Kaya, 2006 suggests it needs to be $\sqrt{N}$, with N being the simulation size.

[1]The first two dimensions comes from the generation of the two driving Brownian motion in the Heston model, and the last comes from the correlation between the dynamics.

4 Quasi-Monte Carlo

In this chapter we will discuss the QMC method, which is often referred to as the deterministic Monte Carlo method. The QMC method makes use of low discrepancy sequences, which as the name of the method suggest are quasirandom sequences, where we in the normal Monte Carlo method, apply pseudorandom sequences. The more evenly the sequence is distributed, the higher accuracy is achieved when calculating the numerical integration problem. We will later see how the QMC convergence is superior to Monte Carlo by using the Koksma-Hlawka inequality. Empirically in literature we often observe that convergence rate be faster than the theoretical upper bound even i1n high dimensions. As in Monte Carlo the QMC integration problem is similar

$$\int_{[0,1]^d} f(\mathbf{x})dx, \tag{4.1}$$

where d determines the dimension. Here the QMC method approximates this integral with

$$\int_{I^d} f(\mathbf{x})dx \approx \frac{1}{N}\sum_{i=1}^{N} f(u_i), \tag{4.2}$$

Note that equation (4.1) is a closed d-dimensional unit (hyper)cube. This formally is similar to the Monte Carlo estimation, but $u_1, ..., u_N$ comes from a low discrepancy sequence, instead of a pseudorandom generator.

Low Discrepancy sequence

A more rigorous study of low discrepancy sequences can be found in Niederreiter, 1992. To attain the highest success rate for the value of every single underlying asset price simulated in a sequence, we employ low discrepancy sequences, instead of the pseudorandom sequences. Also, to measure uniformity we make use of the definition of discrepancy. We consider discrepancy as a tool to measure deviations of empirical distribution. Under the

uniform distribution the value will theoretically be given as the volume of the subset of the uniform distribution with a respective measure defined within the volume of the subset $[0,1]^d$.

Definition 1 (Discrepancy). Given a sequence $u_1, ..., u_N \in I^d$, its discrepancy is

$$D(u_1, ..., u_N) = \sup_{M \in \mathcal{M}} \left| \frac{A\{u_i \in M\}}{N} - \lambda_d(M) \right|,$$

here λ_d is the Lebesgue measure, essentially the volume of the set M. $A\{u_i \in A\}$ denotes a counting function that indicates the quantity of elements of n with $1 \leq n \leq N$ for which $u_n \in M$. Finally $\mathcal{M}$ is a collection of sets in the form of

$$\prod_{j=1}^{d} [c_j, d_j), \quad 0 \leq c_j \leq d_j \leq 1.$$

Definition 2 *The star discrepancy:* $D_N^*(P) = D_N^*(y_1, ..., y_N)$ *of the point set* P *is defined as*

$$D_N(P) = D_N(\mathcal{J}^*; P),$$

where $\mathcal{J}^*$ *is the family containing all subintervals of* I^d *formulated as* $\prod_{j=1}^{d} = [0, u_j]$.

If we take supremum over the collection of sets $\mathcal{M}^*$ of the form

$$\prod_{j=1}^{N} [0, d_j], \qquad 0 \leq d_j \leq 1.$$

It will result in the star discrepancy

$$D^*(u_1, ..., u_N) = \sup_{M \in \mathcal{M}^*} \left| \frac{A\{u_i \in M\}}{N} - \lambda_d(M) \right|.$$

On the pages 23-24 in Niederreiter, 1992 shows the following properties of the one-dimensional star discrepancy and discrepancy

$$D*(u_n, ..., u_N) \geq \frac{1}{2N}, \quad D(u_1, ..., u_N) \geq \frac{1}{N},$$

where both discrepancy terms reach their minimum at midpoint, $u_i = \frac{2_i - 1}{2N}$, $i = 1, ..., N$. Nevertheless, these points will not characterize the first N points of an endless sequence. Thus a set of points with $N + 1$ terms is a set of unrelated points.

Under the lenses of numerical integration we very much would like to increase N in a

simulation by making use of the previous $N-1$ terms of the sequence, without the need to generate the same sequence again as that would be superfluous work and dampen the computational speed.

From the works of Niederreiter, 1992 we also obtain that the lower bound for the d-dimensional star discrepancy function

$$D*(u_1, ..., u_N) \geq c_d \frac{\log(N)^{d-1}}{N}, \quad \forall n > 1,$$

where the constant c_d depends solely on the dimension of the sequence. Henceforth, we use the term "low discrepancy" for sequences that have attained star discrepancy of $\mathcal{O}(\log^d N/N)$. Note that for a fixed dimension, the logarithmic numerator term will be dominated by the denominator term.

Introduction to (t, m, d)-Nets and (t, d)-Sequences

One of the earliest attempts to construct low discrepancy sequences, was the one-dimensional Van der Corput sequence. The construction of the Van der Corput sequence demonstrates the main building blocks to create low discrepancy sequences. In this chapter we follow in the footsteps of Niederreiter, 1992 on how we define (t, m, d)-nets and a simple example of such a net. The implementation has been expanded to higher dimensions using sequences such as Sobol, Halton, Faure and others. The sequences have in common that they are based on the prime bases in the other dimensions. As we are constructing low discrepancy sequences, we will be using these prime bases, hence we use simple intervals which are partitions of our domain with respect to the same base we use to create low discrepancy sequences.

Definition 3 *Let $d \geq 1, b \geq 2$, then let a subinterval E of $[0, 1)^d$ be on the form*

$$E = \prod_{i=1}^{d} \left[a_i b^{-d_i}, (a_i + 1) b^{-d_i} \right),$$

for integers d_i, a_i, with $d_i \geq 0$ and $0 \leq a_i < b^{d_i}$ for $1 \leq i \leq d$ is the construction of the so-called elementary interval in base b.

A finite sequence of b^d points in $[0, 1)^d$ will be a $(0, m, d)$-net in base b, if for all elementary interval in base b of volume b^{-m} contains only one point of the sequence. In general what separates (t, m, d)-net and a (t, d) sequence, is if the sequence of points $X_1, X_2, ...$ is finite number of points or infinite number of points.

Example 4 If we look at the 1-dimensional case ($d = 1$), then obviously $d_1 = 1$, and let $b = 3$, then elementary interval under the prime base of b will be: $[0, 1/3)$, $[1/3, 2/3)$, $[2/3, 1)$. If we instead had chosen $d_1 = 0$, based on a definition of elementary intervals we would have the unit interval $[0, 1)$.

Example 5 In a 2-dimensional case ($d = 2$), where we let the base $b = 2$ and let $d_1, d_2 = 0$, then $b_1, b_2 = 0$, which generates the following partition, $[0, 1) \times [0, 1)$. If then $d_1, d_2 = 1$, then we obtain the following intervals,

$$\begin{cases} [0, 1/2] \times [0, 1/2] \\ [0, 1/2] \times [1/2, 1) \\ \vdots \end{cases}$$

Definition 6 *Allow $0 \leq t \leq m$ where t, m are integers. A (t, m, d)-net in the base if b is a set of points P of a^m points in $[0, 1)^d$ such that $A(E; P) = b^t$ for all elementary interval E within base b with $\lambda_d(E) = b^{t-m}$.*

Definition 7 *Allow $t \geq 0$ and be an integer. Given a sequence $x_0, x_1, ...$ of points in the interval $[0, 1)^d$ will be ordained as the so-called (t, d) sequence in base b if, $\forall k \geq 0$, $k \in \mathbb{Z}$ and $m > t$, the point set consisting of the x_n with $kb^m \leq n < (k + 1)b^m$ is minted the (t, m, d)-net in base b.*

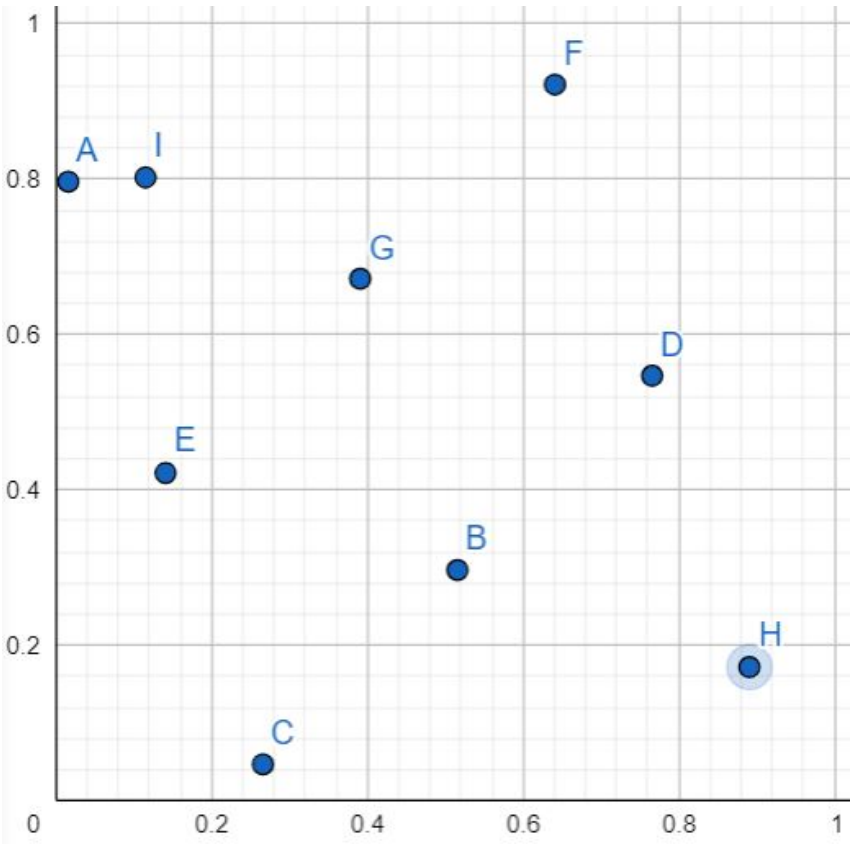


Figure 4.1: Example sketch of a $(0, 3, 2)$-net

The definitions for specifically a (t, m, d)-net and (t, d) sequences are given in Niederreiter, 1992 with more elaboration. All elementary interval in base b of volume, b^{t-m}, contains exactly b^t points of the sequence. Thus, the volume of b^{t-m} is equivalent to the ratio $\frac{b^t}{b^m}$, in the aggregated amount of points.

The distinction between a (t, m, d)-net and (t, d)-sequences is infinity, a sequence will have an infinite amount of points with good uniformity, whereas the net is a finite set of points with a good uniformity. Since it is easier to distribute a finite set evenly in space rather than fiddling to make an infinite sequence, this provides a good uniformity for all the finite segments of the space it is filling out.

For example purposes, take a $(0, d)$ Faure sequence, if we were to take a finite point set from this particular sequence as follows $\{q_k : jb^m \leq k \leq (j+1)b^m\}$ (in this context $j = 4$), then we produce a $(0, m, d)$-net. Again b is the base of the sequence, which relies on the dimension d. Figure (4.1) is an example of a $(0, 3, 2)$-net, where the dimension is equal to two, and m is equeal to three. We obtain this from a 2-dimensional Faure sequence with a base of 2, by extraction of elements from the Faure Sequence with indices

4×2^3 through $5 \times 2^3 - 1$.

The Sobol Sequence

One of the first quasirandom low discrepancy sequences is the Sobol sequence. It was published back in 1967 by Russian mathematician Ilya Meyerovich Sobol in Sobol, 1967. The Sobol sequence is the first construction of a so-called (t, d)-sequence. It begins with a Van der Corput sequence, but maintain its trademark with the exclusive use of base equal to 2. The remaining coordinates are of a d-dimensional Sobol sequence constructed by permutations of some segments of the Van der Corput sequence. Therefore, every element is created from the same construction, but applying a non-identical generator matrix G in base 2. In the generator matrix we will denote the columns as $g_i, i = 1, ..., m$ are the directional coordinates in base 2 (directional coordinates are coefficients of many primitive polynomials). The size of the generator matrix will be dependent on how high a dimension we operate in, and the length of binary depiction of the n'th element of the sequence.
A thorough explanation for the algorithm to generate a Sobol sequence, can be found in Bratley and Fox, 1988. We will briefly outline the major points: If we want to generate the jth component of the points in the Sobol sequence, we will need a choice of primitive polynomial of some degree s_j in the field $\mathbb{Z}_2 = \{0, 1\}$,

$$x^{s_j} + a_{1,j}x^{s_j-1} + a_{2,j}x^{s_j-2} + ... + a_{s_j-1,j}x + 1 \tag{4.3}$$

where the coefficients $a_{1,j}, a_{2,j}, ..., a_{s_j-1,j}$ are either 0 or 1. Then we construct a sequence of positive integers $m_{1,j}, m_{2,j}$ by the following recurrence interaction

$$m_{k,j} := 2a_{1,j}m_{k-1,j} \oplus 2^2 a_{2,j}m_{k-2,j} \oplus ... \oplus 2^{s_j-1}a_{s_j-1,j}m_{k-s_j+1,j} \oplus 2^{s_j}m_{k-s_j,j} \oplus m_{k-s_j,j}. \tag{4.4}$$

Here the operation $\oplus$ is the bit-by-bit exclusive-or operator (XOR). The starting values of $m_{1,j}, m_{2,j}, ..., m_{s_j,j}$ are freely chosen as long as the following condition met: for each $m_{k,j},\ 1 \leq k \leq s_j$, is an odd integer and less than 2^k. The infamous direction numbers for the Sobol sequence are defined as follows

$$v_{k,j} := \frac{m_{k,j}}{2^k}. \tag{4.5}$$

Then we define $x_{k,j}$, as the ith component of the ith point in a Sobol sequence, as given by

$$x_{i,j} := i_1 v_{1,j} \oplus i_2 v_{2,j} \oplus ..., \tag{4.6}$$

here i is written as $i = (...i_3 i_2 i_1)_2$, implying that i_k is the kth digit from the right. The notation $(\cdot)_2$ signifies that it is a binary representation of a number.

Gray Code Implementation

To generate the N-bit Sobol numbers using the Gray code implementation as in Antonov and Saleev, 1979 the following algorithm is used

1 A primitive polynomial of any degree s_j is picked, and we have that $a_0 = a_d = 1$,

$$x^{s_j} + a_{1,j}x^{s_j-1} + a_{2,j}x^{s_j-2} + ... + a_{s_j-1,j}x + 1.$$

2 Then we need to create an array of odd integers of the elements $m_{k,j}$, where the only condition is $1 \leq m_{k,j} \leq 2^k$ and needs to be odd.

$$m_{k,j} := 2a_{1,j}m_{k-1,j} \oplus 2^2 a_{2,j}m_{k-2,j} \oplus ... \oplus 2^{s_j-1}a_{s_j-1,j}m_{k-s_j+1,j} \oplus 2^{s_j}m_{k-s_j,j} \oplus m_{k-s_j,j}.$$

3 Then we create the directional numbers $v_{k,j}$ as follows

$$v_{k,j} := \frac{m_{k,j}}{2^k}.$$

4 Then the jth component for the ith Sobol point, $x_{k,j}$

$$x_{i,j} := g_{i,1}v_{1,j} \oplus g_{i,2}v_{2,j} \oplus ...,$$

where $g_{i,k}$ is the kth from the right of a Gray code of i in binary, i.e., $\text{gray}(i) = (...g_{i,3}g_{i,2}g_{i,1})$. This step is equivalent to equation 4.6 see Antonov and Saleev, 1979.

For a more comprehensive understanding of the algorithm and methods used, see Joe and Kuo, 2003,Bratley and Fox, 1988 and Antonov and Saleev, 1979.

Sobol Sequence Results

In the plots below we have taken advantage of the function, `sobol` in the MATLAB toolbox `Statistics and Machine Learning` to generate the Sobol sequence scatter plot, and for our pseudorandom sequence we have used `rand` in MATLAB.

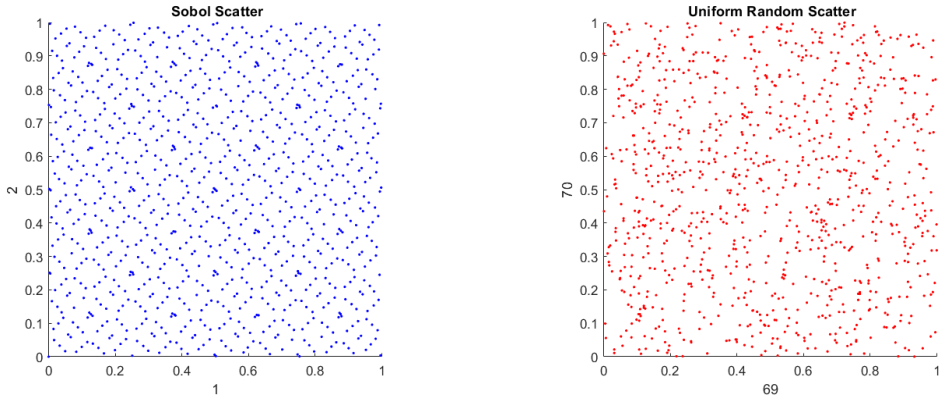


Figure 4.2: Here we have plotted an orthogonal projection (2D image) of the first 1000 points of a Sobol sequence (Left side) and pseudorandom sequence (Right side).

We can extrapolate from figure (4.2) that the Sobol sequence has a more systematic way of filling the cubed space versus the random function. From the left panel we observe in the Sobol sequence a couple of minor clusters in some of its square formations, thus making it a less uniform spread over the unit cube. The right panel depicts pseudorandom points, which are relatively random; there are some areas where we are let with spaces larger than the Sobol sequence, and we observe some heavier clustering, which is to be expected from a pseudorandom sequence. These results are similar to other papers using different programming language and/or algorithms. There is also a great animation of this example, which can be found following this link https://wsiegenthaler.github.io/lobos/web-example.html (Siegenthaler, 2020), which slowly adds more points in the cubed space, and visually shows how the Sobol sequence fills out the space at a better pace than the pseudorandom sequence (example is based on the paper Joe and Kuo, 2008). With higher order projections we will also see in general that most low discrepancy sequences deteriorate with the order of the dimensions.

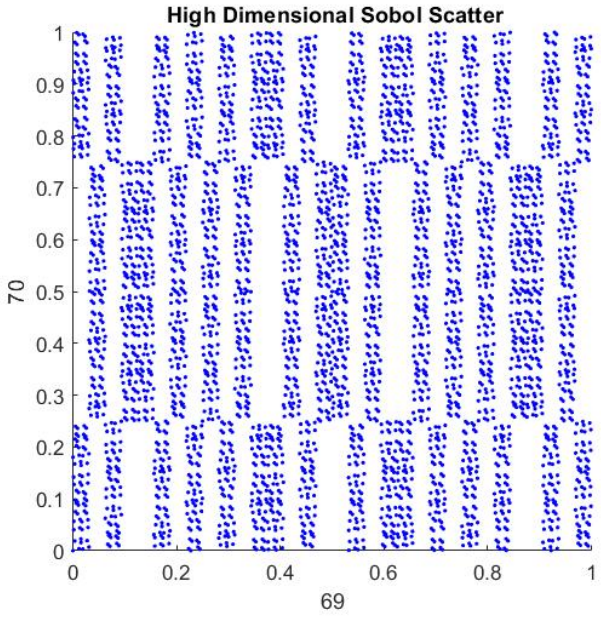


Figure 4.3: An orthogonal projection (2D image) of a 69th dimension versus the 70th dimension of the Sobol sequence. The sample size was 4096 points

In the figure (4.3) above see an almost chessboard-like structure of the Sobol sequence distribution in higher dimensions. Compared to the figure (4.2) we are left large open spaces, implicating that the discrepancy therefore is higher than the low dimensional cases. In general, all sequence see some kind of deterioration in higher dimension, but seen in the article Morokoff and Caflisch, 1994, the Halton sequence and the Faure sequence also have the same deterioration in higher dimension, instead of a chessboard-like structure, they obtain a linear band-like structure. In results from Fox, 1986 and Bratley and Fox, 1988 they saw the Sobol sequence being multiple times faster than Halton, Faure and pseudorandom sequences. Note that their simulation was done on the super computer Cyber 855 and Cyber 800 respectively, and they were comparing with an optimized Sobol sequence, which is not exactly an apples to apples comparison.

In the paper by Morokoff and Caflisch, 1994, they did a similar comparison without optimizing any of the algorithms for the low discrepancy sequences, and saw that the random number generator was faster than the low discrepancy sequences. Nonetheless, it was quite evident from both papers that the accuracy was higher with the low discrepancy sequences, which we also tested with our algorithms in Appendix A.

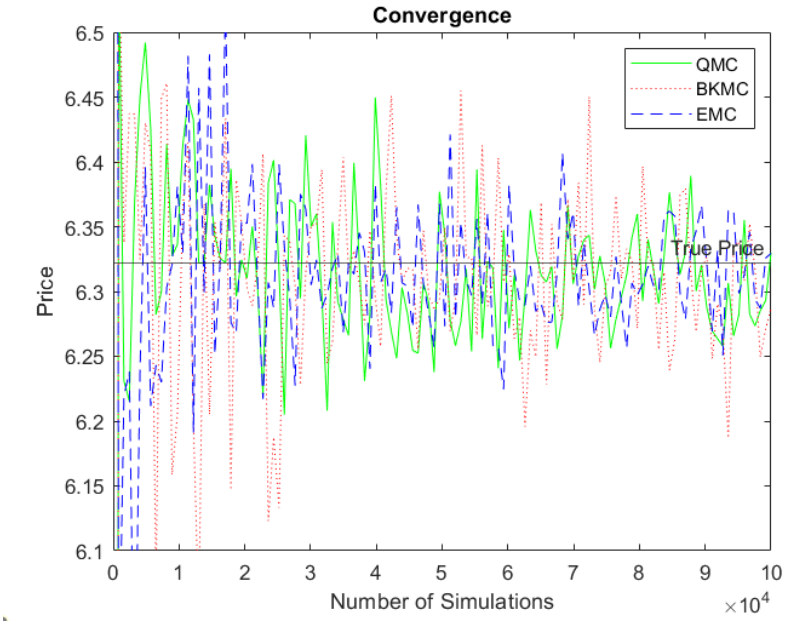


Figure 4.4: In this figure we have added QMC price estimates of a European option. Option parameters are the same as in figure (3.1).

Luck strikes again as we see from the figure (4.4) that QMC method also converge towards the true option price as the simulation increases. From this figure there is a visible improvement of using the quasirandom numbers (QMC) vs the pseudorandom numbers (BKMC). There is not much literature comparing these two empirically, as the above figure (4.4) depict a better performance from the QMC versus the BKMC in the earlier stage, but becomes more equal as more simulations are added. Important remark; in our algorithm for the Broadie-Kaya scheme applied here, the distinguishing factor is the use of pseudorandom numbers and quasirandom numbers. Note that the EMC is still better than both methods in terms of stability and how quickly it converges, albeit the QMC method is not far off,

From theory it is not clear which method should perform better in practise between the EMC and QMC. From the implementation done in Broadie and Kaya, 2006 where they only state to have used a Euler scheme for their Monte Carlo method[1], it was clear from their tables that the QMC method performed better in terms of estimate, but their time estimate to only be better when sampling beyond 2,5 mio. samples, which in my perspective defeats the purpose of this approach. The idea behind the QMC method was to show its superiority in better convergence to estimates, thus allowing for a smaller simulation size. This was also the intention behind applying the pseudorandom number to

[1]It is not clear from their article, according to Lord et al., 2010 it was the absorption scheme, which is not the best performing scheme according to their article

the Broadie Kaya method - to see whether the convergence would require a larger number of simulation to obtain the same estimates as the QMC method. It is difficult to know if our code requires more optimization, as the EMC method does outperforms the QMC method in terms of price convergence and simulation time.

We will elaborate on the estimates of price later on in this paper, and compare the results to the article Broadie and Kaya, 2006, to get a better understanding of the performances of the different methods.

Next we will look into the computational speed of both methods, where in general for the same number of simulations we should see the same runtime for both the QMC and BKMC methods.

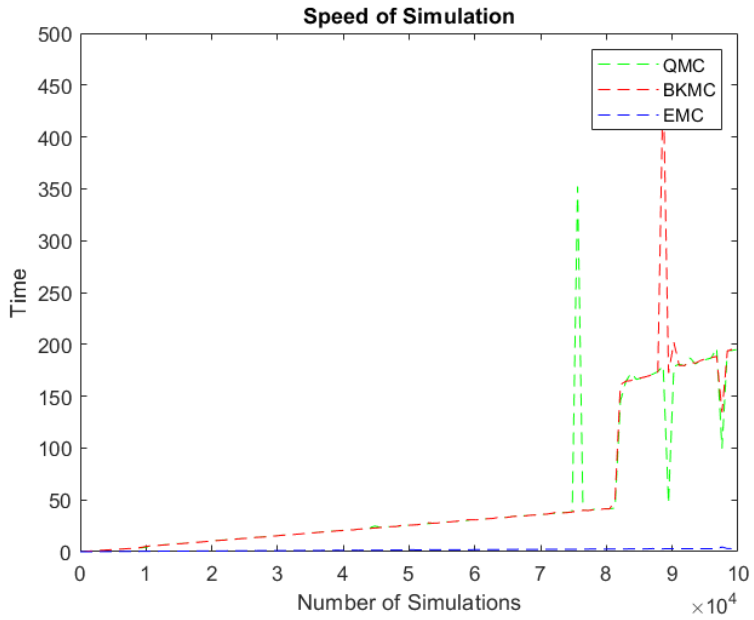


Figure 4.5: Plot of computation time measured in seconds as a function of simulations done. Again we have 124 data points of the time estimates of the different methods. Same option parameters as figure (3.1).

As suspected the runtime is very similar and should have been linear, but the consequences of using a laptop computer is that they tend to overheat, and the computer was running at night to lessen the burden on the processor. This is why we observe spikes; we did however run the code again, as can be viewed in the appendix. This is furthermore why it is not possible to increase the simulation size in a similar fashion as in Broadie and Kaya, 2006, - since the same computer that is running the code is need throughout day, hence it introduces bad estimates on the speed of calculation, as the extra workload overheats the computer, and more time needed to allow for a cooldown before continued

computation.

Randomized QMC

Randomized QMC (RQMC) is the application of randomization (scrambling) on the low discrepancy sequences. There are many benefits to scrambling the low discrepancy sequences; one is to remedy the deterioration in discrepancy in higher dimensions as seen in Joe and Kuo, 2008. This will consequently improve our estimation of the prices we obtain. The major benefit to scrambling is that it lays out a practical method to acquire the error estimate from our QMC method, simply by treating each scrambled sequence as an independent sample from a family of randomly scrambled quasirandom numbers, as seen in Owen, 1995. Ergo, RQMC surmounts one of the biggest obstacles for QMC, while preserving the same convergence rate as QMC, which is desirable as the calculation of the upper bound in Koksma-Hlawka inequality. The RQMC method still retains the good uniformity property of the low discrepancy sequences, but also gives us the statistical error analysis. RQMC starts with the same integral

$$I_{\text{RQMC}} = \int_{[0,1)^d} f(x)dx$$

Approximated with the sums of the form

$$Q_n({}^r q) = \frac{1}{n}\sum_{n=1}^{n} f({}^r q_n)$$

where ${}^r q_n$ in this case is a family of d-dimensional randomized quasirandom sequence indexed with a random parameter r. RQMC preserves many of the great properties of the previous Monte Carlo methods

$$\mathbb{E}[Q_n({}^r q)] = I_{\text{RQMC}}, \quad \text{Var}(Q_N({}^r q)) \leq \mathcal{O}\left(\frac{\log(N)^{d-1}}{N}\right), \quad |Q_N({}^r q) - I| \leq V(f)D_N^*({}^r q). \tag{4.7}$$

These properties are carried over from the QMC method and Monte Carlo method, as RQMC retains its properties from QMC and regains some from standard Monte Carlo as described in Ökten and Willyard, 2010.

Scrambling

Art B. Owen introduced the random scrambling method back in 1995 (see his paper Owen, 1995). The method introduced by Owen originally applies unique permutation on each elementary interval to fully randomize the (t, s)-sequence. It can help to imagine the set of all elementary intervals as a binary tree, with the most crucial digit being the upper most interval of the tree, and the following digit selecting a subinterval. The permutation of a single digit will start a wave that forcefully rearranges the subinterval, but does not change the number of points in any of the intervals.[2]

Using this method is expensive in practical computations; it requires large number of independent permutation (we would have to store information in the size of db^K, where K digits er permuted, b is the base of the digits and d dimension), which leads to an excessively large size of memory storage needed to store the information. In order to avoid the collection of data and computation, Matoušek proposed various approximations to Owen scrambling.

Most quasirandom sequences are built upon a parameter named the base of the sequence. Let the base be b, then the nth element of the sequence $q_n = (q_n^{(1)}, ..., q_n^{(d)})^\top$, is calculated by an algorithm that is given the base b expansion of the integer n and then outputs a real number $q_n^{(i)}, i = 1, ..., d$. Let $\mathbf{q}$ be a d-dimensional sequence in base b

$$\mathbf{q} := \begin{pmatrix} q_n^1 \\ \vdots \\ q_n^d \end{pmatrix}, n = 1, 2, ...$$

and now let

$$^r\mathbf{q} := \begin{pmatrix} {}^r q_n^1 \\ \vdots \\ {}^r q_n^d \end{pmatrix}, n = 1, 2, ...$$

We obtain ${}^r q_n^i$ from q_n^i. Let the base b expansions of these numbers be

$$q_n^i = (0.a_1 a_2 ... a_k)_b$$

[2]Owen named the method nested uniform scrambling, but it became known as the Owen scrambling.

$$^{r}q_n^i = (0.c_1c_2...c_k)_b$$

In Matoušek, 1998, he proposes a random digit scrambling method, for each dimension i we randomly generate k independent permutations $\pi_1, ..., \pi_k$ of $\{0, 1, 2, ..., b-1\}$ and set

$$c_j = \pi_j(a_j),\ j = 1, ..., k$$

In order to obtain independent copies of random digit scrambled sequences, we randomly generate other sets of independent permutations $\pi_1, ..., \pi_k$ of $\{0, 1, 2, ..., b-1\}$ for each $i = 1, ..., d$ and go through the calculations again. To obtain this result we implement the left matrix scramble followed by a random digital shift[3]. Let $\mathbf{L}_1, ..., \mathbf{L}_d$, be a non-singular lower triangle $k \times k$ matrices and $\mathbf{e}_1, ..., \mathbf{e}_d$, be an $k \times 1$ vector. Thus, a random linear scrambling with a digital shift,

$$\begin{pmatrix} c_1 \\ c_2 \\ \vdots \\ c_k \end{pmatrix} = \begin{pmatrix} l_{1,1} & 0 & 0 & ... & 0 \\ l_{1,1} & l_{1,2} & 0 & ... & 0 \\ \vdots & & & & \\ l_{k,1} & l_{1,2} & l_{1,3} & ... & l_{1,k} \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ \vdots \\ a_k \end{pmatrix} + \begin{pmatrix} e_1 \\ e_2 \\ \vdots \\ e_k \end{pmatrix}. \tag{4.8}$$

Here the diagonal components $l_{i,i}$ of the lower-triangular matrix are generated stochastically from a uniform distribution on $\{1, 2, ..., b-1\}$ and the terms $l_{i,j}$ for $i < j$ are stochastically generated from a uniform distribution on $\{0, 1, ..., b-1\}$. The entries in vector $\mathbf{e}$, are also generated at random from a uniform distribution on $\{0, 1, ..., b-1\}$. Important to note is that all arithmetic operation in equation (4.8) are done in modular arithmetic b.

This is Matoušek's two way to approximate the Owen scrambling method, random linear scrambling and random linear digit scrambling. We see from Owen, 1995 that this randomization preserves the properties of the (t, m, d)-net and (t, d) sequence (to see proof of each property see Owen, 1995).

1 **Equidistribution**:If A_i is a (t, m, d)-net in base b then X_i is a (t, m, d)-net in base b with probability 1.

2 **Uniformity**:Let $A \in [0, 1)^d$ and let X be the result of the random scrambling described above applied on A. Then X has a uniform distribution on $[0, 1)^d$. Mathematically for any Lebesgue measurable $G \subseteq [0, 1)^d, P(X \in G) = \lambda_d(G)$, the d-dimensional Lebesgue measure of G.

[3]This can intuitively be understood as a random linear map and shift operation

Sobol Scrambling

We will be using the same toolbox mentioned for `sobol`; using `sobolset` and `scramble` with the specific option `'MatousekAffineOwen'`, which allows us to use the faster version of Owen scrambling. A more in depth explanation can be found in Owen, 1995 and Hong and Hickernell, 2003.

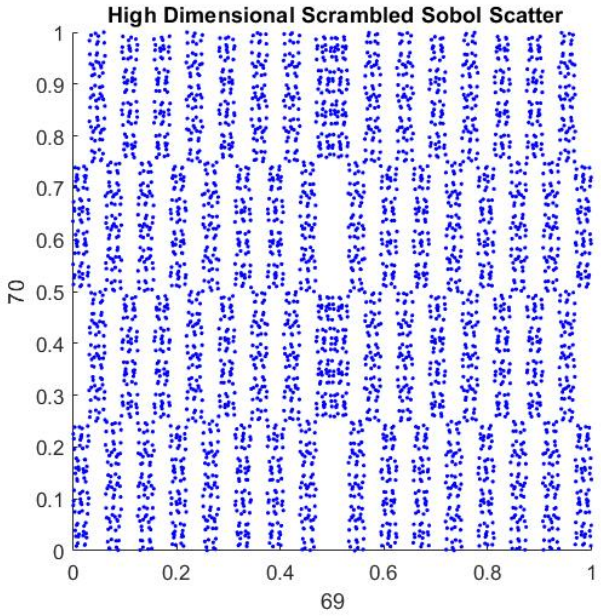


Figure 4.6: Showcases a 2D-projection of a 69th dimension versus the 70th dimension of a Owen scrambled Sobol sequence. The sample size was 4096 points

Comparing this to the non-scrambled Sobol sequence, we see a similar chessboard-like pattern. The pattern has become finer; if continue the chessboard comparison, there are more spaces than previous, meaning the discrepancy has become lower. Thus the accuracy should be improved based on these plots. When running the code for the plots of different option pricing method the following two option have been added in the function `sobolset`, `Leap` and `Skip`[4].

`Leap`, which allows us to specify the number of points in the sequence we leap over and omit for every point taken. The standard leap value is 0, which corresponds to taking every point from the sequence. Depending on how the Leap value is chosen, it is possible to improve the quality of your point set, but if poorly chosen it is plausible to completely ruin the uniformity of the quasirandom sequence, see Kocis and Whiten, 1997.

[4]So all plots that have used quasirandom sequences have had these two options added

`Skip`, which allows us to skip a cetain number of initial points in the sequence to omit from the point set, specified as a positive integer scalar. We will remark that the, initial points of a quasirandom sequence may exhibit unfortunate properties. In many cases the first points will often be $(0, 0, 0, ...)$, and when the counterpart of the sequence is $(1, 1, 1, ...)$, which is not often seen, will cause the sequence to be quite unbalanced. Lastly some points can starting points may manifest correlation among different dimension, while it has disappeared later on in the sequence. Initial points of a sequence sometimes exhibit undesirable properties. For example, the first point is often (0,0,0,...), which can cause the sequence to be unbalanced because the counterpart of the point, (1,1,1,...), never appears. Also, initial points often exhibit correlations among different dimensions, and these correlations disappear later in the sequence. See Appendix A for a deeper study on these two options as we here will just post that for all generations on the Sobol sequence to estimate options prices we have used the following `Skip` = 100 and `Leap` = 1000 with Matoušek approximated version of Owen scrambling results in the following

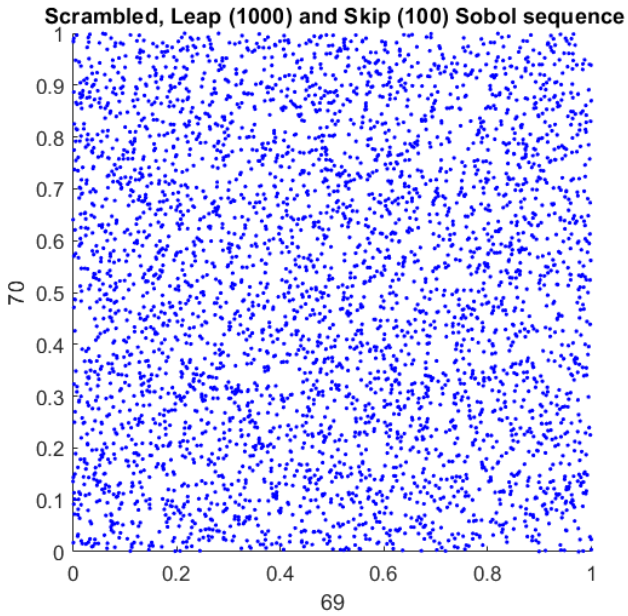


Figure 4.7: We have plotted the option prices for a European option for the EMC, QMC and lastly but not least the RQMC as a function of simulations, parameters are the same as figure (3.1).

And from this we see that these two options are very attractive as they kill the chess-like pattern we saw previously and therefore the points are more uniformly distributed than before, which is the desired property.

In the next plot we have added the last Monte Carlo method to our comparison scheme,

being RQMC method against the QMC and EMC.

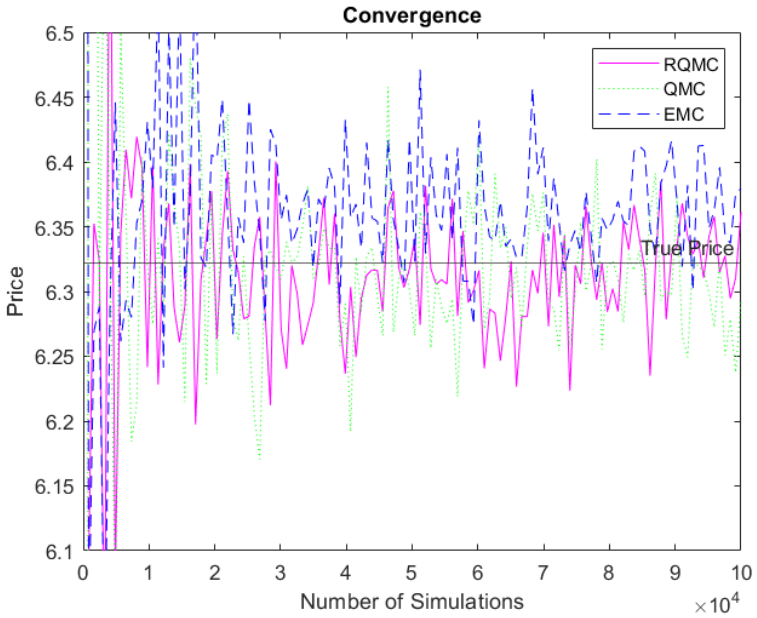


Figure 4.8: We have plotted the option prices for a European option for the EMC, QMC and lastly but not least the RQMC as a function of simulations, parameters are the same as figure (3.1).

Again with the similar setup as before, we again observe an increase in performance of the RQMC method compared to the QMC. We observe a better convergence towards the price, lowered oscillation levels after 20 thousand simulations for the RQMC method. However when comparing the RQMC method with the EMC, we see that there is potential for the RQMC method, however both in convergence rate and stability of oscillation level the RQMC is being beaten. Note that we still only used Euler discretization with Full Truncation in the EMC method, which not only easier to implement, but also a quicker method in terms of speeds.

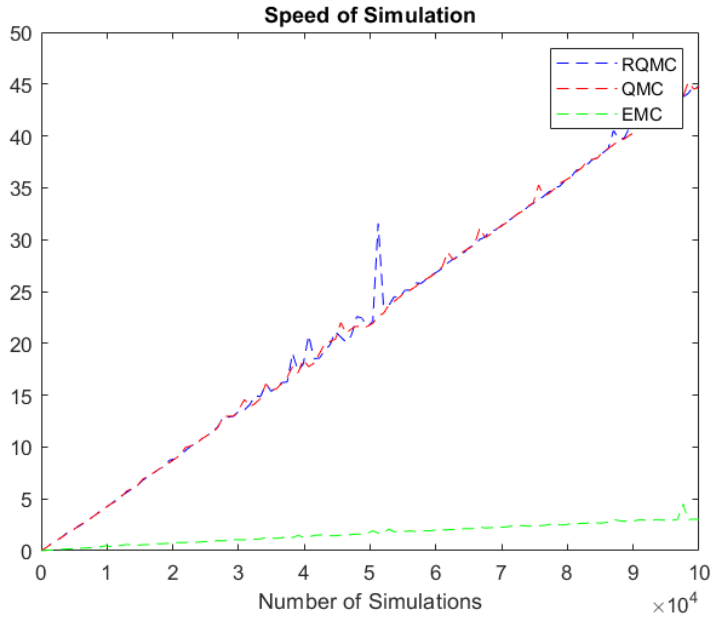


Figure 4.9: We have computed the option price of a European option and this is a plot of the running time for the EMC, QMC and RQMC as a function of simulations, parameters are the same as figure (3.1).

As suspected the times estimate for all the MC methods using pseudorandom sequences and quasirandom sequences be identical, as long as they are running the same amount of simulation. This is a result of the method doing the same computation for each simulation, thus the methods will never be faster in time using the same number of simulation with our implementation of the algorithm.

Bounds of Quasi-Monte Carlo

In this section we will delve into the so-called roof of QMC, which is the error bound for the QMC method. The popularity of Quasi-Monte Carlo integration stems from its successful implementation in financial problems, risk management and so on. The convergence rate of the standard Monte Carlo simulations emphasizes the CLT to obtain a probabilistic convergence rate of $\mathcal{O}(N^{-1/2})$, contrary to the QMC integration method, which boasts an upper error bound of $\mathcal{O}(\frac{\log(N)^d}{N})$. This error bound for the QMC method has been made well-known by the Koskma-Hlawka inequality. The Koskma-Hlawka inequality uses the assumption of the integrand having a finite variation.

Intriguing results from the application of QMC has shown that it is not rare to observe the numerical convergence rate as high as $\mathcal{O}(N^{-1})$ for different problems. The empirical results made by Paskov and Traub, 1996 and Paskov, 1997 showed that the QMC method

is an efficacious instrument when calculating for some high dimensional integrals.

We briefly mention the works of Hardy, 1906 and Krause, 1903, as they have made some of the widely used definitions on QMC error bounds. In many articles mentioning QMC, we will see the papers pointing out that the variation of function is bounded in a Hardy-Krause sense. These references are for the definition made by Hardy and Krause in their study of double Fourier series. The definition of total variation is constructed from their works. We define the variation in the famous Hardy-Krause sense of a function f as in Niederreiter, 1992. We have for a single variable,

$$V[f] = \int_0^1 \left| \frac{\partial f}{\partial t} \right| dt. \tag{4.9}$$

Now in the d-dimensional case, we use the convenient definition mentioned in Niederreiter, 1992,

$$V^{(d)}[f] = \int_0^1 ... \int_0^1 \left| \frac{\partial^d f}{\partial t_1 ... \partial t_d} \right| dt_1 ... dt_d, \tag{4.10}$$

holds as long as the indicated partial derivatives are continuous on the interval $I^d = [0,1]^d$. Let $1 \geq k \geq d$ and $1 \geq i_1 < i_2 < ... < i_k < d$, where we have $V^{[k]}(f; i_1, ..., i_k)$ as our variation in the so-called "sense of Vitali". Then with this as a restriction of f on the kth dimension $\{t_1, ..., t_d \in I^d : u_j = 1\}$ for $j \neq i_1, ..., i_k$. then

$$V(f) = \sum_{k=1}^{i} \sum_{1 \geq i_1 < i_2 < ... < i_k < d} V^{[k]}(f; i_1, ..., i_k)$$

is the famous variation of f on I^d in the sense of Hardy and Krause. Therefore, if f is of bounded variation in the Hardy-Krause sense, it will imply that $V(f)$ is finite. With this criteria of variation, we can finally construct the inequality in Hlawka, 1961 also famously known as the Koksma-Hlawka inequality.

Theorem 8 *If f has bounded variation $V(f)$ on I^d in the sense of Hardy-Krause, then for any $x_1, ..., x_N \in [0,1]^d$, we have*

$$\left| \int_{[0,1]^d} f(x)dx - \frac{1}{N} \sum_{i=1}^{N} f(x_i) \right| \leq V(f)^d D(d)(x_1, ..., x_N), \tag{4.11}$$

where $D(d)(x_1, ..., x_N)$ is the so called star discrepancy of a d-dimensional low discrepancy sequence.

In Neiderreiter's more recent work, Niederreiter, 2003, he established a more general framework for error bounds with an arbitrary probability space and under the assumption of a specific uniformity trait for the point set. His error bound have no requirement of the Koksma-Hlawka inequality. X. Jiang and J. R. Birge have found some fruition from this error bounds framework of Niederreiter and derived an asymptotical error bound, $\mathcal{O}(N^{-1/2})$ for specifically the European put option pricing.

We will look into this at a later point of the dissertation. For now, we will look more into the approach that Papageorgiou took in his two articles Papageorgiou, 2001 and Papageorgiou, 2003. In these two papers he describes a finite dimensional numerical integration dilemma; in Papageorgiou, 2001 he considers an integrand as a function of a Euclidean norm, and in Papageorgiou, 2003 he considers an even more general class of function in the integral than previously.

As in Papageorgiou, 2001 we look at an approximation of a weighted high dimensional integral,

$$I_{d,g} = \int_{\mathbb{R}^d} f(g(x))\phi_d(x)dx, \tag{4.12}$$

again d is the dimension, we have $\phi_d(x) = (2\pi)^{\frac{-d}{2}} e^{\frac{-||x||^2}{2}}$ as the Gaussian weight. Here we assume that the given continuous function, $g : \mathbb{R}^d \to \mathbb{R}$, and $f : \mathbb{R} \to \mathbb{R}$ such that $I_{d,g}(f) < \infty$. The function g will also be assumed to have the probability measure defined as follows,

$$\mu(B) = \int_{\mathbb{R}^d} \mathbb{1}_B(g(x))\phi_d(x)dx, \quad B \in \mathcal{B}(\mathbb{R}). \tag{4.13}$$

Here B is a borel measurable and ϕ_d is the d-dimensional Gaussian weight. This is equivalent to a Lebesgue measure. We then do a change of variable in equation 4.12 to reduce the problem to a 1-dimensional problem. We define Ψ as a cumulative normal distribution (CDF) with mean zero and variance of one[5] and we write the equation (4.12) to an equivalent integral over cube, the $[0,1]^d$,

$$I_{d,g}(f) = \int_{\mathbb{R}^d} f(g(x))\phi_d(x)dx = \int_{[0,1]^d} f(g(\Psi^{-1}(t_1), ..., \Psi^{-1}(t_d)))dt.$$

For all deterministic points $t_i = (t_{i,1}, ..., t_{i,d}) \in (0,1)^d$, with $i = 1, ..., n$, we construct the points $x_i = (x_{i,1}, ..., x_{i,d}) \in \mathbb{R}^d)$ with $i = 1, ..., n$ by allowing $x_{i,j} = \Psi(t_{i,j})$ where $i = 1, ..., n$

[5]Here we have defined $\Psi(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{y^2}{2}} dy, \quad x \in [-\infty, \infty]$

and $j = 1, ..., d$. This would allow us to approximate the integral from (4.12) by the QMC method,

$$I_{d,g,n}(f) = \frac{1}{n}\sum_{i=1}^{n} f(g(x_i)). \tag{4.14}$$

In derivation of the error equation and also the convergence rate for the method $I_{d,g,n}$, the following conditions are needed with 4 classes of functions F_1, F_2, F_3, and F_4.

Definition 9 *We let x_0, γ, and β be in $\mathbb{N}^+$. We assume the function g to have tails of measure μ that fulfils the following inequalities:*

$$1 - \mu(x) \leq \frac{\gamma}{e^{\beta x^2}},\ x \geq x_0 \tag{4.15}$$

$$\mu(-x) \leq \frac{\gamma}{e^{\beta x^2}},\ x \geq x_0 \tag{4.16}$$

where $\mu(x) := \mu(-\infty, x),\ x \in \mathbb{R}$.

Now we let F_1 be a class of functions where $f : \mathbb{R} \to \mathbb{R}$, such that $I_{d,g}(f) < \infty$, f is absolutely continuous, f' exist a.e., and

$$\left[\int_{\mathbb{R}} \left(\frac{f'(x)^2}{e^{\alpha|x|}}\right)\right]^{\frac{1}{2}} \leq A.$$

Here A and α are given positive numbers.

Now we let F_2 be a class of functions that is defined similarly to F_1 though there is a small exception, that is the condition above is modified as such,

$$ess\sup\left\{\frac{|f'(x)|}{e^{\alpha|x|}} : x \in \mathbb{R}\right\} \leq A.^{6}$$

Definition 10 *The classes for F_3 and F_4 follow a very identical setup. Let x_0, γ, β and δ be in $\mathbb{N}^+$. We assume the function g to have tails of measure μ that fulfils the following inequalities:*

$$1 - \mu(x) \leq \frac{\gamma}{e^{\beta(\log(\delta x)^2}},\ x \geq x_0 \tag{4.17}$$

$$\mu(-x) \leq \frac{\gamma}{e^{\beta(\log(\delta x)^2}},\ x \geq x_0 \tag{4.18}$$

Notice here that the tails of μ are different than in definition (9). The tails will degrade

[6]Where ess sup is the generalization to some measurable functions of the maximum. In technical terms the difference is that values of functions on a set of measure 0 will not alter the essential supremum. Example, a given measurable function $f : Y \to \mathbb{R}$, where Y is the measure space with measure μ, the ess sup is the smallest number a such that the set $\{y : f(y) > a\}$ has measure 0. If no number exists, as in the case of $f(y) = \frac{1}{y}$ on $y \in \{0, 1\}$ then the essential supremum is ∞

proportionally with $\frac{1}{x^{\beta(\log(x)}}$.

Now let F_3 *be the class of function* $f : \mathbb{R} \to \mathbb{R}$*, implying that* $I_d(f) < \infty$*, also* f *is absolutely continuous,* f' *exists a.e., and*

$$\int_{\mathbb{R}} |f'(x)|dx \leq A,$$

again A *is given and a positive number.*

Lastly F_4 *is defined like* F_3 *with the exception that the above condition is exchanged with (similarly with definition (9)),*

$$ess\sup\{|f'(x)| : x \in \mathbb{R}\} \leq A$$

The bounds derived from the problem in equation (4.12), are based on the different classes of the function F_1, F_2, F_3 and F_4. As seen above we can group together the classes 1 and 2 as they are characterized by their limitations on the integrand and supremum by the term $f'(x)e^{-\alpha}$. The classes 3 and 4 are then characterized by the limits set by the term $\int_{\mathbb{R}} |f'(x)|dx \leq A$ and $\operatorname{ess\,sup}\{|f'(x)| : x \in \mathbb{R}\} \leq A$ respectively.

We introduce some notation on the definition for discrepancy under a generalized arbitrary probability space from Papageorgiou, 2003. Let μ be a probability measure on $\mathbb{R}$, let $x_i \in \mathbb{R}$, $i = 1, ..., n$ be any points and $n > 1$. Thus, the difference of the theoretical distribution and the empircal distribution is given by a discrepancy function

$$R_\mu(M) = \frac{A(M; N)}{N} - \mu(M), \qquad E \subset \mathbb{R}_+, \tag{4.19}$$

where $A(M; N)$ is the number u_i contained in M and is not dependent on μ, but on the points in u_i. The discrepancy of the points $u_1, ..., u_N$ with regards to the probability measure μ is defined as follows,

$$D_{\mu,N} = D_{\mu,N}(u_1, ..., u_N) = \sup_M |R_\mu(M)|, \tag{4.20}$$

where the supremum is taken over all the subset on the form $M = [0, u),\ u \in \mathbb{R}_+$.

Error & Convergence of Quasi-Monte Carlo

We will define the integration error by

$$\epsilon(I_{d,N}, f) = |I_d(f) - I_{d,N}(f)|.$$

Here the worst case error for such a class of function will be described by,

$$\epsilon(I_{d,N}) = \sup_{f \in F} \epsilon(I_{d,N}, f).$$

In Papageorgiou, 2001 it is also shown how the upper bound of the error has an order $\mathcal{O}(\sqrt{\log(N)}/N)$, with a rather impractical constant term. Therefore it is not easy to compute said constant term of the upper bound, as it relies on several factors $e(I_{d,1}), d\Gamma(d/2)$ and $\eta(\mu) = \int_{\mathbb{R}^+} \left\| \mathbb{1}_{[0,r)}(\eta) - \mu(r) \right\| dr$, where $\eta = \mu^{-1}(0.5)$.

So if we take the class of functions F_3, the worst possible error would be bounded by

$$\epsilon(I_{d,g,N}, F_3) \leq A \sup_{u \in \mathbb{R}} |R_\mu(u)| = A D_{\mu,N},$$

where $D_{\mu,N} = D_{\mu,N}(g(u_1), ..., g(u_N)),\ g(u_i), u_i \in \mathbb{R}^d,\ i = i, ..., N$, is the discrepancy with respect to our measure μ. Note that minimizing the discrepancy $D_{\mu,N}$ will lower the integration error in the QMC.

Given a sample set made of function evaluations at points $x_i \in \mathbb{R}^d,\ i = 1, ..., n$, it can be shown that the error is highly dependent on the discrepancy $D_{\mu,n}$, with respect to μ, of the points $g(x_i) \in \mathbb{R}$

Lemma 11 *Consider the integral in (4.12) and assume that at least one of the following definitions in definition 9 or definition 10 have their conditions fulfilled. Let $x_i \in \mathbb{R}^d,\ i = 1, ..., n$ be any points, $n \geq 1$. Then for $f \in F_i = 1, ..., 4$ we have*

$$I_{d,g}(f) - I_{d,g,n}(f) = \int_{\mathbb{R}} R_\mu(x) f'(x) dx,$$

where $R_\mu(x) = R_\mu(x; g(x_1), ..., g(x_n))$,

Proof: We will proceed in the same manner as chapter 2 in Niederreiter, 1992, when determining the error of the QMC method, approximating the integral of a differentiable

function. For $n, d \geq 1$ and $x_i \in \mathbb{R}^d,\ i = 1, ..., n$ we have

$$\begin{aligned}
&\int_{\mathbb{R}^d} f(g(x))\phi_d(x)dx - \frac{1}{n}\sum_{i=1}^{n} f(g(x_i)) \\
&= \int_{\mathbb{R}} f(t)\mu'(t)dt - \frac{1}{n}\sum_{i=1}^{n} f(g(x_i)) \\
&= \int_0^1 f(\mu^{-1}(1))dt - \frac{1}{n}\sum_{i=1}^{n} f(g(x_i)) \\
&= \int_0^1 y(s)ds - \frac{1}{n}\sum_{i=1}^{n} y(u_i), \quad y = f \circ \mu^{-1}, u_i = \mu(g(x_i)) \\
&= \int_0^1 R(t)'(t)dt, \quad R(t) = \frac{1}{n}\sum_{i=1}^{n} 1_{[0,t]}(u_i) - t \\
&= \int_0^1 R(t)\frac{df(z)}{dz}|_{z=\mu^{-1}}(\mu^{-1})'(t)dt \\
&= \int_0^1 R(t)\frac{df(z)}{dz}|_{z=\mu^{-1}}\mu^{-1}(t) \\
&= \int_{\mathbb{R}} f'(t)R(\mu(x))dx \\
&= \int_{\mathbb{R}} f'(t)R_\mu(x)dx,
\end{aligned}$$

and the proof is complete. ■

We now look into a class of functions F, where we defined the worst error event of the method $I_{d,g,n}$ by

$$e(I_{d,g,n}, f) = \|I_{d,g}(f) - I_{d,g,n}(f)\|, \quad f \in F.$$

$$e(I_{d,g,n}, F) = \sup_{f \in F} e(I_{d,g,n}, f)$$

The two equations above define integration errors and worst case error for a class of functions F, respectively.

Proposition 12 *For classes of function $F_i,\ i = 1, ..., 4$, under the conditions of definition 9 and definition 10, the error of the method $I_{d,g,n}$ in equation (4.14) satisfies*

$$e(I_{d,g,n}, F_i) = \sup_{f \in F_i} \int_{\mathbb{R}} |f'(x)R_\mu(x)|dx, \quad i = 1, ..., 4$$

and

$$e(I_{d,g,n}, F_i) \leq A\sqrt{\int_{\mathbb{R}} e^{\alpha|x|}R_\mu^2(x)dx},$$

$$e(I_{d,g,n}, F_i) \leq A \int_{\mathbb{R}} e^{\alpha|x|} |R_\mu(x)| dx,$$

$$e(I_{d,g,n}, F_i) \leq A \sup_{x \in \mathbb{R}} |R_\mu(x)| = M D_{\mu,n},$$

$$e(I_{d,g,n}, F_i) \leq A \int_{\mathbb{R}} |R_\mu(x)| dx$$

Proof: From lemma 11 we have that $e(I_{d,g,n}, F_i) \leq \sup_{f \in F_i} \int_{\mathbb{R}} |f'(x) R_\mu(x)| dx,\ i = 1, ..., 4.$ Thus for $f \in F_i$ we need to consider an according function w, such that $w'(x) = f'(x) \cdot \text{sign}(f'(x), R_\mu(x))$ a.e., where we also need that $\text{sign}(x) = \pm 1$ when $x < 0$ and $x \geq 0$, respectively. Then we have $w'(x) R_\mu(x) \geq 0$ a.e., w is absolutely continuous and $w \in F_i$, allowing us to deduce

$$e(I_{d,g,n}, F_i) = \sup_{f \in F} \int_{\mathbb{R}} |f'(x) R_\mu(x)| dx, \quad i = 1, ..., 4.$$

Then it becomes rather apparent for the last inequalities. First, we take the definition of F_1 and apply the Cauchy-Scwharz inequality to the integral above, while the other three follows from the definition of F_2, F_3 and F_4 respectively. ■

Corollary 13 *If $\lambda \gg \mu$ and has support on the bounded or semi-bounded interval, then the proposition 12 holds by substituting the domain of integration by the support of μ, supp(μ). Furthermore, if supp(μ) is bounded, Proposition 12 implies*

$$e(I_{d,g,n}, F_i) = \mathcal{O}(D_{\mu,n}), \quad i = 1, ..., 4$$

We have seen the error of QMC is dependent on discrepancy, $D_{\mu,n} = D_{\mu,n}(g(x_1), ..., g(x_n))$, with the points $g(x_i),\ x_i \in \mathbb{R}^d,\ i = 1, ..., n$, with respect to the measure, μ. Thus minimizing the discrepancy of $D_{\mu,n}$ reduces the QMC error.

In theorem 5.1 of the paper Papageorgiou, 2003, he showcases that the upper error bound for F_1, F_2, F_4 are of the order $\mathcal{O}\left(\frac{1}{\sqrt{N^{1+\zeta \log(N)}}}\right)$, where ζ is a different constant term and $\zeta \geq 0$. For the class F_3 we see that the error of the upper bound is of the order $\mathcal{O}\left(\frac{1}{N}\right)$. This order of error we need the discrepancy of the sample points used in the QMC method to satisfy $D_{\mu,n} = \mathcal{O}(n^{-1})$, in other words it must decay with that order. However, that is not possible since we have no way, currently, to construct a d-dimensional low discrepancy sequence satisfying that assumption.

Error Bounds Under a General Probability Space

A more general error bound was presented in Niederreiter, 2003, Theorem 5.1 and Theorem 5.2 using the so-called $(\mathcal{M}, \mu)$-uniform sets

Theorem 14 *Again let $(\Omega, \mathcal{B}, \mu)$ be a random probability space and let $\mathcal{M} = \{M_1, ..., M_k\}$ be divisions of Ω with $M_j \in \mathcal{B}$ for $1 \leq j \leq k$. Then for any $(\mathcal{M}, \mu)$-uniform set $\mathcal{P} = \{\omega_1, ..., \omega_N\}$ and any bounded μ integrable function f on Ω we have*

$$\left| \sum_{n=1}^{N} f(x_n) - \int_{\Omega} \right| \leq \max_{1 \leq j \leq k} \left(\sup_{t \in M_j}(t) - \inf_{t \in M_j}(t) \right)$$

Theorem 15 *Again let $(\Omega, \mathcal{B}, \mu)$ be a random probability space and let $\mathcal{M} = \{M_1, ..., M_k\}$ be divisions of Ω with $M_j \in \mathcal{B}$ for $1 \leq j \leq k$. Then for any $(\mathcal{M}, \mu)$-uniform set $\mathcal{P} = \{\omega_1, ..., \omega_N\}$ and any bounded μ integrable function f on Ω we obtain*

$$\left| \frac{1}{N} \sum_{n=1}^{N} f(\omega_n) \int_{\omega} f d\mu \right| \leq \max_{1 \leq j \leq k} \sum_{j=1}^{k} \mu_j(M_j) \left(\sup_{t \in M_j}(t) - \inf_{t \in M_j}(t) \right)$$

Our approach is based on Theorem (14), of Niederreiter, 2003, which is a generalization of the Koksma-Hlawka inequality to abstract probability spaces and point sets. Theorem (15) applies to any $(\mathcal{M}, \mu)$-uniform point set, independently of how the set was constructed (such as using Sobol sequence or similar). The following example uses the same approach to bound on an European option's prices as the paper Jiang and Birge, 2004.

Example 16 (European put and call option). The European put option's price in the log-normal model has the following form

$$e^{-rT} \int_0^1 \left(K - S_0 \exp\left[\left(r - \frac{\sigma^2}{2} \right) T + \sigma \sqrt{T} \Phi^{-1}(t) \right] \right)^+ dt.$$

In this example K, S_0, r, T, σ are positive constants, Φ is a standard normal CDF and recall the notation $(x)^+ = \max(0, x)$. Notice that for a specific t the integrand will become

zero

$$\begin{aligned} 0 =& K - S_0 \exp\left[\left(r - \frac{\sigma^2}{2}\right)T + \sigma\sqrt{T}\Phi^{-1}(t)\right] \\ \frac{\log(K)}{\log(S)} =& \left(r - \frac{\sigma^2}{2}\right)T + \sigma\sqrt{T}\Phi^{-1}(t) \\ \Phi\left(\frac{\frac{\log(K)}{\log(S)} - \left(r - \frac{\sigma^2}{2}\right)T}{\sigma\sqrt{T}}\right) =& t \end{aligned}$$

Thus the integrand will become 0 if $t > \hat{t}$, where

$$\hat{t} = \Phi\left(\frac{\ln\left(\frac{K}{S_0}\right) - \left(r - \frac{\sigma^2}{2}\right)T}{\sigma\sqrt{T}}\right)$$

and leads to the following integral

$$e^{-rT}\int_0^{\hat{t}}(K - S_0\exp\left[\left(r - \frac{\sigma^2}{2}\right)T + \sigma\sqrt{T}\Phi^{-1}(t)\right]dt.$$

Consider now the $(\mathcal{M}, \mu)$-uniform point set $\mathcal{P} = \{x_1, x_2, ..., x_N\}$, where μ is our uniform Lebesgue measure on $X = \{0, 1\}$, and let $\mathcal{M} = \{M_1, ..., M_k\}$ be partitions of $[0, 1)$. Now let $f = e^{-rT}(K - S_0\exp\left[\left(r - \frac{\sigma^2}{2}\right)T + \sigma\sqrt{T}\Phi^{-1}(t)\right]$, we will notice that $0 \leq f(t) \leq e^{-rT}K$ for $0 < t < \hat{t}$, and f is monotonically decreasing with respect to t, and $f'(t)$ is bounded on $(0, \hat{t})$. Furthermore by Theorem (14) we obtain,

$$\left|\sum_{n=1}^{N} f(x_n) - \int_0^1 f dt\right| \max_{1\leq j\leq k}\left(\sup_{t\in M_j} f(t) - \inf_{t\in M_j} f(t)\right).$$

Specifically, pick $k = N$ such that the intervals $M_j = \left[\frac{(j-1)}{N}, \frac{j}{N}\right), j = 1, ..., N$ and then for some $\xi \in [0, \hat{t})$ will give use the

$$\sup_{t\in M_j} f(t) - \inf_{t\in M_j} f(t) = f\left(\frac{j-1}{N}\right) - f\left(\frac{j}{N}\right) = -\frac{1}{N}f'(\xi) = \frac{C}{N},$$

Where C is some positive constant, using the fact that f is monotonically decreasing from $(0, \hat{t})$ and that f' is bounded. This means that the QMC error bound for European put options is $\mathcal{O}\left(\frac{1}{N}\right)$. Thus from the put-call parity, the European call option has the same error bound. This bound is better than the bound found in Jiang and Birge, 2004, which is $\mathcal{O}\left(\frac{1}{N^{0.5+\varepsilon}}\right)$, for any $\varepsilon > 0$, which is not a better bound than the one derived here.

A brief investigation on the error bounds obtained from Monte Carlo and QMC shows how

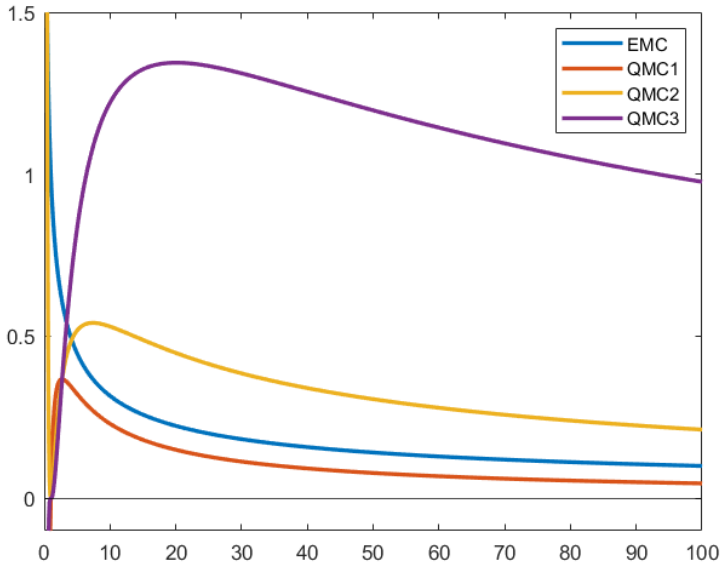


Figure 4.10: The number on QMC label indicates the dimension that we are determining the error size of $d = \{2, 3, 4\}$ in $\frac{\log(n)^{d-1}}{n}$.

the error size becomes with the increase to simulation numbers N. We left out the error bound order of $\frac{1}{N}$, as it is currently not viable option, thus any analysis or comparison is pointless. This figure (4.10) shows the peculiar behaviour of an upper bound of the QMCs' bounds, for the first n number of simulation. The bound is in fact increasing for all dimension of $\frac{\log(n)^{d-1}}{n}$ for the first n simulation. Note that it quickly changes as n increases, and for these three bound becomes strictly decreasing after $n = 20$. Furthermore for $d = 2$, we observe that QMC1 is actually strictly below the Monte Carlo bound, where $d = 3$ needs $N \geq 5504$ for it to be lower and for $d = 4$ we need $N \geq 2.15 \cdot 10^7$. This shows that for N large enough we can be sure that QMC error will be lower, but as we see in the next chapter it converges at an even faster rate in our case.

5 Numerical Results

In this section we will compare some of the numerical results found in Broadie and Kaya, 2006. Our primary focus will be on the EMC, QMC and RQMC method from our paper. The simulation experiments in this and other sections of our paper were executed by a laptop computer with an intel core i7-10750 2.60 GHz processor and 16 GB RAM, running windows 10. The code was written in MATLAB programming language and compiler was MATLAB R2021b.

Before we begin the comparison, we observe how our algorithms prices European call options that are "out of the money", "at the money", and "in the money", it gives us a quick view on how the algorithm handles options that are in different states, which is comparable to the real world.

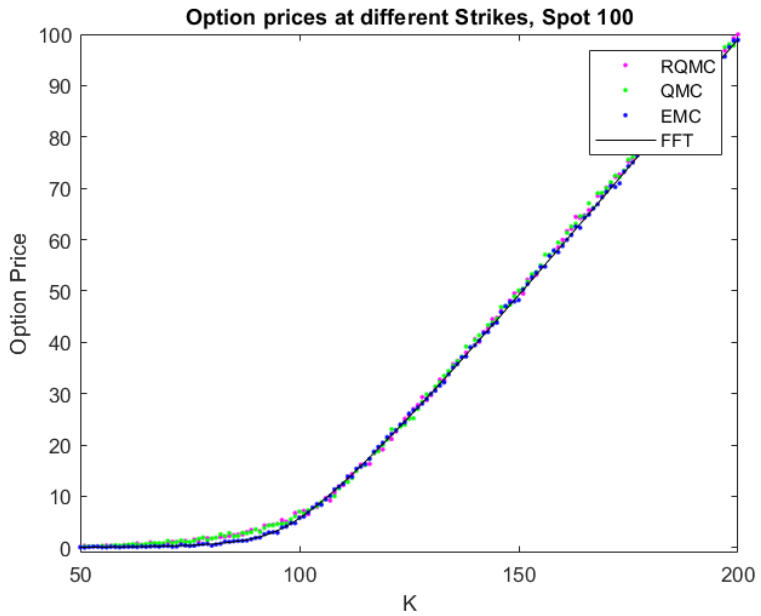


Figure 5.1: Four different schemes estimates of the European call option, using strikes going from $K = 1, ..., 200$, for computational speed only a 1,000 simulation was used to estimate each price.

Our figure demonstrates that our algorithms follow the classical tendencies for price

of a European call option as the price increases. Overall the method follows the true price calculated with the FFT[1]. Please note that we only used 1,000 sample paths for computational speed - with this in mind we still observe that only the RQMC and QMC are overestimating the prices of a European option, which are "out of the money". This is definitely not ideal, as the EMC method appears to be following the price better, even with 1,000 simulations.

We also wish to numerically compare these schemes in a different manner, beyond their given price estimate and runtime. We work in the Heston model, in which exists a semi-closed solution to obtain the price of a European option. Thus, we will compare EMC, QMC and RQMC methods mainly through their RMSE, but for EMC and RQMC it is possible to look upon the standard error (SDE) and bias. If we let α' be the estimator for the derivative price obtained through simulation and α be the true price. Then the SDE, bias and RMSE is given as follows

$$\text{SDE} := \frac{\sqrt{\mathbb{E}[(\alpha' - \mathbb{E}[\alpha'])^2]}}{\sqrt{N}}, \qquad \text{Bias} := |\mathbb{E}[\alpha'] - \alpha|, \quad \text{RMSE} := \sqrt{\text{SDE}^2 + \text{Bias}^2} \tag{5.1}$$

In the tables (5.1), (5.2) and (5.3) we will observe numerical results of the different schemes on an European call option. The first set of tables will be using the same parameters for the European call option as seen throughout the dissertation. The star after the number indicates if it was the best (smallest) value obtained through the simulation.

[1]We followed the paper Carr and Madan, 1999, for our algorithm and how we determine the price in difference cases of parameters in the Heston model

Table 5.1: EMC

No. of simulation	No. of time steps	Bias	SDE	RMSE	Run time (sec.)
10,000	100	0.0825	0.0901	0.1221	0.3786*
40,000	200	0.0392	0.0454	0.0600	1.3661*
90,000	300	0.0236	0.0304	0.0385	2.3942*
490,000	700	0.0015	0.0130	0.0131	330.09*

Table 5.2: QMC

No. of simulation	RMSE	Run time (sec.)
10,000	0.0179*	4.2069
40,000	0.0078*	17.809
90,000	0.0060*	42.909
490,000	0.0030	346.74

Table 5.3: RQMC

No. of simulation	Bias	SDE	RMSE	Run time (sec.)
10,000	0.0305*	0.0189*	0.0359	4.2440
40,000	9.9817e-04*	0.0077*	0.0078*	17.729
90,000	4.9817e-04*	0.0065*	0.0067	41.761
490,000	2.2738e-04*	0.0020*	0.0021*	344.85

From these tables the major link is the value of the RMSE, therefore we start the comparison here. We are able to observe from the tables that the RMSE values in general are many times lower for QMC and RQMC compared to the EMC. We see that QMC's RMSE value is around 6-8 times lower, while RQMC in general is around 4-8 times lower. Recall that RMSE value is a measure of the average model prediction error, it explains how well our method predicts the true value. Thus, QMC and RQMC have 4-8 times smaller errors on their option price predictability, meaning their accuracy of estimate compared to EMC method is a lot higher. Note that we utilized a better version of the EMC method than the one used in Broadie and Kaya, 2006, and we observe similar results with the RMSE values being 4-8 times lower for QMC and RQMC method versus the better version of the EMC method. This next plot also shows this difference in accuracy at the cost of time.

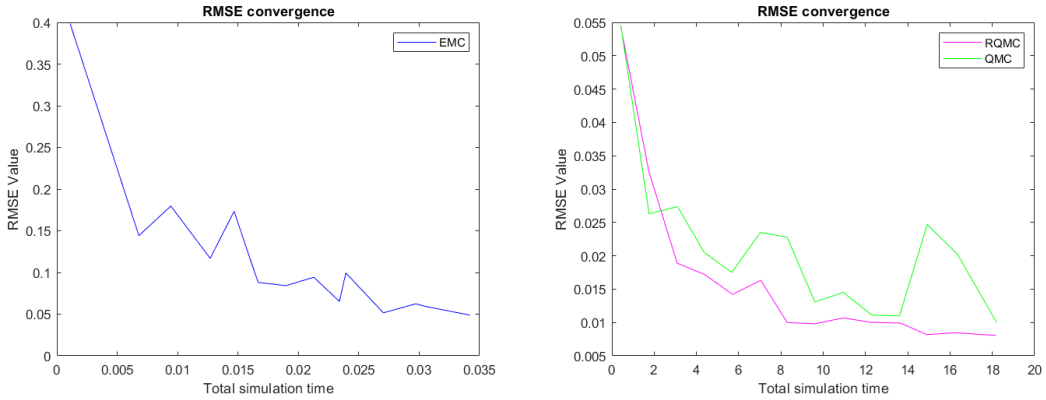


Figure 5.2: Here we have three methods plotted based on their RMSE value as a function of their total simulation time measured in seconds. Option parameters are the same as in figure (4.4). The simulations spans from 1,000-40,0000 and the EMC method has a stepsize of 100.

The methods RQMC and QMC, appear to follow a similar pattern in terms of the RMSE value, though in the first trial with 10,000 sample path RMSE values is twice as high for RQMC compared to QMC. For the rest of the trials we see near identical values, meaning they are equal in terms of their prediction error.

For the two other estimates, bias and SDE of RQMC and EMC, we see that RQMC is clearly outperforming in both categories by a fair bit. The bias in the first trial for the RQMC has a lower bias by factor of 2 and in the second trial with 40 thousand paths, we see a factor difference close to 40 times lower. This difference persist throughout the rest of the trials, albeit not nearly as earth-shattering as the second trial.

And the SDE story is very similar to the RMSE values, thus we will cut it short, by saying that the RQMC SDE values were a couple of times lower, when compared to the EMC method.

Overall we see that RQMC and QMC wins all categories except speed for most of the trial cases. Note that it is not possible to calculate the bias for the QMC method, thus the SDE and RMSE values are the same for that method specifically. Thus RQMC and QMC are equal in everything disregarding the bias category. This picture of the performance is very similar to the one described in Broadie and Kaya article, even with the improved version of the EMC method. Although the speed in EMC is better the estimate is lacking in comparison, to improve on that, we could increase the stepsize of the EMC method as seen in table 3 in Broadie and Kaya, 2006. Note that such an increase will lead to decrease in the computational speed.

The option parameter used in table 2 by Broadie and Kaya, 2006 are $S = 100$, $K = 100$, $V_0 = 0.09$, $\kappa = 2.00$, $\sigma = 1.0$, $\rho = -0.3$, $r = 5.00\%$ and $T = 5$. Thus, allowing us to more easily compare the tables and also comment upon our algorithm performance versus theirs.

Table 5.4: EMC

No. of simulation	No. of time steps	Bias	SDE	RMSE	Run time (sec.)
10,000	100	0.7163	0.5911	0.9287	0.0895*
40,000	200	0.5750	0.2831	0.2889	0.3629*
90,000	300	0.2698	0.1909	0.3302	2.3821*
160,000	700	0.0321	0.1457	0.1492	124.72

Table 5.5: QMC

No. of simulation	RMSE	Run time (sec.)
10,000	0.0937*	4.7922
40,000	0.0474*	19.4312
90,000	0.0343*	45.5300
160,000	0.0271*	77.17

Table 5.6: RQMC

No. of simulation	Bias	SDE	RMSE	Run time (sec.)
10,000	0.0923*	0.0865*	0.1265	4.7788
40,000	0.0601*	0.0486*	0.0773	19.5069
90,000	0.0485*	0.0361*	0.0605	45.3059
160,000	0.0248*	0.0244*	0.0348	77.01*

Comparing our RMSE values with the one obtained in Broadie and Kaya, 2006, we see that our EMC, QMC RQMC beats their respective opponents in their article. Not only are we able see that Euler with Full Truncation is better than the allegedly Euler with absorption. We also see that the both QMC and RQMC is an improvement to their method. Although the QMC method did outperform the RQMC method in RMSE for all trials. This is slightly unexpected, since the method should be somewhat equal, therefore it could be shifting which one performed the best. If you look to figure (7.2) it is very clear that it is possible for the standard Sobol sequence to have a lower error than the randomized one. The only thing where our algorithm is worse than Broadie and Kaya is in terms of speed, which would be even more evident if we had used the same simulation size as them.

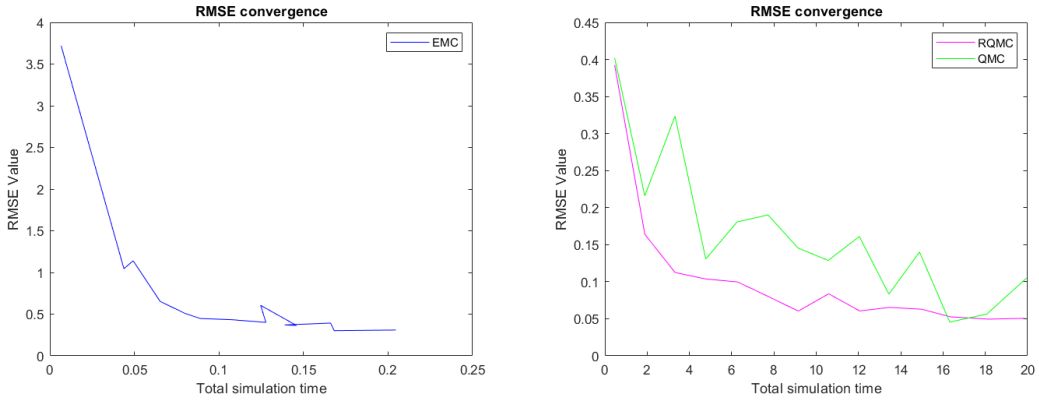


Figure 5.3: Here we have three methods plotted based on their RMSE value as a function of their total simulation time measured in seconds. Option parameters are the same as in table (5.4). The simulations spans from 1,000-40,0000 and the EMC method has a stepsize of 100.

Figure (5.3) is for the second case of option parameters described earlier, here we see that adding 5 years to the maturity does increase the error size by around a factor of 10 for all three methods. Nonetheless the picture remains the same, QMC and RQMC follow each other quite nicely, although QMC is a bit more erratic we see in general that RQMC remains on the lower side og the spectrum. EMC method is around 10 times higher in error size, but the computational speed we see, it might have been an idea to increase the stepsize used.

In summarization, we see that for large enough simulations size that the QMC and RQMC are more than competitive with the EMC method. In many cases it was beaten as seen above indicating that the QMC and RQMC is indeed better for large enough simulation size N. Furthermore it is not clear from these results that the RQMC method is better from the QMC method looking at the RMSE values, which is the only thing we have as an comparison. Recall that the RMSE method and QMC method from figure (4.9) should have identical time estimates, even if are scrambling the points in RQMC, the function in use is built in MATLAB, implying that the algorithm used there is very well-written and fast. Nevertheless, the RQMC method is still very attractive as the randomization of the quasirandom sequence allows the Monte Carlo statistical properties to come back. Hence we are able to see the quality of estimate through bias and SDE, which is highly desirable and from our tables it would appear that RQMC retains the speed and quality of the price estimation seen in the QMC method.

6 Conclusion

In this paper we tested the exact simulation of the stock and variance process under the Heston model. We applied the Broadie-Kaya scheme, which is based on Fourier inversion techniques and condition arguments. The method is theoretically attractive, but it does need some further improvement to be less limiting in practise. We show how to roughly derive the error bound for QMC $\mathcal{O}\left(\frac{\log(N)^d}{N}\right)$ and how it compares to standard Monte Carlo $\mathcal{O}\left(\frac{1}{\sqrt{N}}\right)$. Our benchmark for the Broadie-Kaya scheme with quasirandom numbers, is the Euler discretization with Full Truncation, which is better than the one used in Broadie and Kaya, 2006. Even with the computational efficiency of the EMC method, it has a rather large bias if not an appropriate stepsize per year is used, which is seen in Lord et al., 2010 and Broadie and Kaya, 2006. The increase in stepsize will consequently lead to a decrease in computational speed. To improve on the QMC method, we introduced appropriate randomization techniques based on linear scrambling and digital shifts, which creates the RQMC method, but retaining the qualities of the QMC.

In our final segment we investigated the numerical performance of the methods EMC, QMC and RQMC. We used different sizes of sample paths to see the robustness and capability of the methods. Our findings showed that the QMC and the RQMC methods outperformed the EMC method in terms of accuracy and convergence towards the true value.

We observed that adding more years till maturity on the stock, slows down computational speed of the EMC method at a much faster rate than the QMC and RQMC methods. In general we verified that the QMC and RQMC method produces similar results such as: size of RMSE value, convergence rate and speed of computation, which was observed through the numerical example section. The exact simulation method is still slower in terms of speed under the Heston model, but more attractive when high accuracy is needed.

7 Appendix

This appendix will partitioned into three, Appendix A will be having empirical results and supporting plots created from our codes. Appendix B will contain many known or more common results such as Ito's lemma, Geometric Brownian motions and more. Lastly Appendix C will contain anything not belonging to the other two categories.

Appendix A

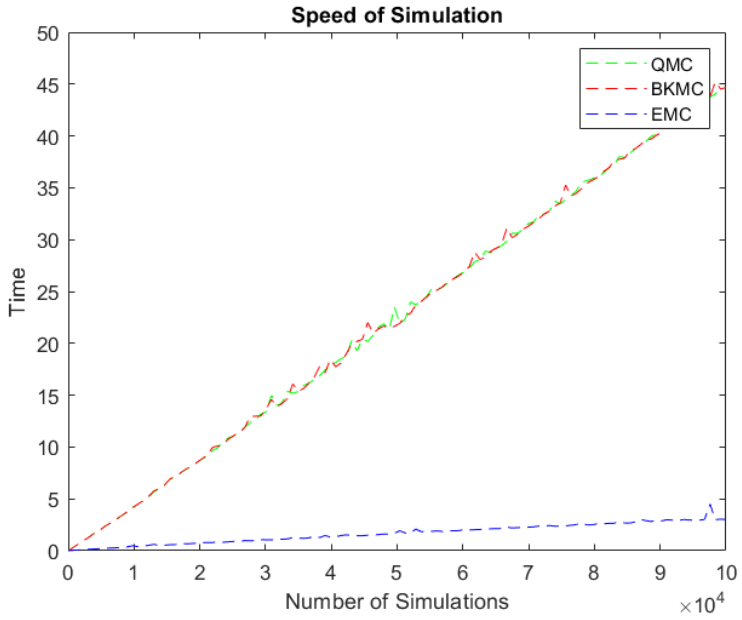


Figure 7.1: Plot of computation time measured in seconds as a function of simulation done. All options same as figure (4.5).

From what we can observe here, the time estimates becomes a lot more linear, with no exorbitant increase in computation time as seen in figure (4.5). This time the laptop did not overheat, as the laptop was allowed to work throughout the night with full fan speed.

7.1.1 Error of Quasi-Random Sequences

In this section we are using a general method to estimate the absolute error for a function $f(x)$ in the interval $[a, b]$ for any of Halton, randomized Halton, Sobol and randomized Sobol Sequence with different values of dimensions. The first plot shows the relation between the absolute error and the dimensions for the function at hand $f(x) = \sin(2x)$[1] in the interval $[0, 1]$.

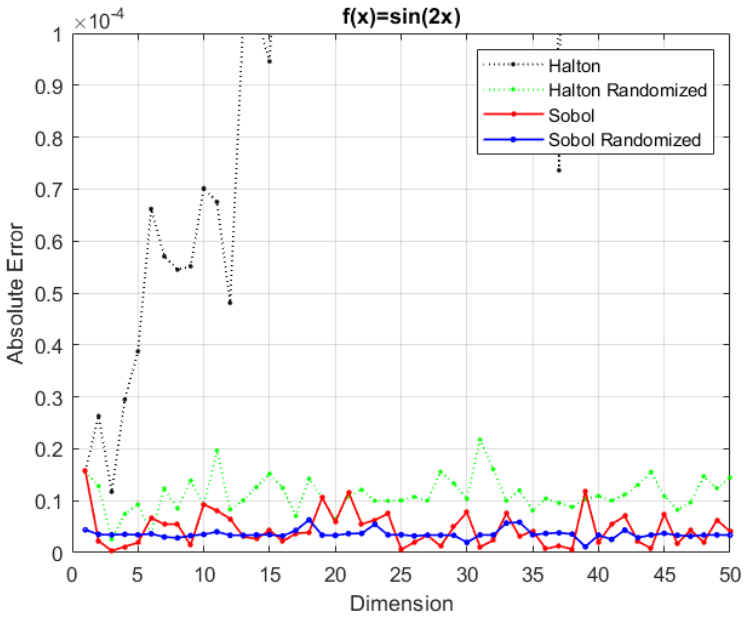


Figure 7.2: Plot of the absolute error of the 4 different sequences. For the randomized Halton sequence the option in MATLAB `RR2`, was used to apply apply reverse-radix scrambling on the Halton sequence. The randomized Sobol sequence used the option `MatousekAffineOwen` described in this paper. We have used 100.000 points for the evaluation of the integral.

The figure (7.2), seen above, is a plot of absolute errors of estimated integrals using Halton and Sobol sequences and a randomized version of them as a function to the dimension of the sequence. Visual inspection of the above mentioned plot shows that the absolute error up to dimension 5 are rather similar for all the sequences. Dimensions $d > 5$ we can clearly see that Sobol sequence and the two randomized sequences are better than the Halton sequence for all values of dimensions. Note that the output here is the average of the absolute sequences, which for the Halton sequence is increasing with the dimension, while for the other three sequences it is rather stable. The randomized Halton sequences does not increase its' absolute error with an increase in dimension, and also became a lot more stable. The Sobol sequence did not see a large decrease in the

[1]There is no reason for this specific function and it was chosen arbitrarily.

absolute error when randomized, but the sequence becomes a lot more stable if randomized.

In the next plot we are looking at the relation between the number of samples and the mean absolute error (MAE) for the function $\exp(-x)^2$ on the interval [0.1]. We are using the MAE to elucidate the effect of increasing the number of samples for evaluation of the integrals, will decrease the error of evaluation. The MAE is the average of the absolute differences between the exact and estimated integrals. Algorithms for both plots can be found in Appendix C.

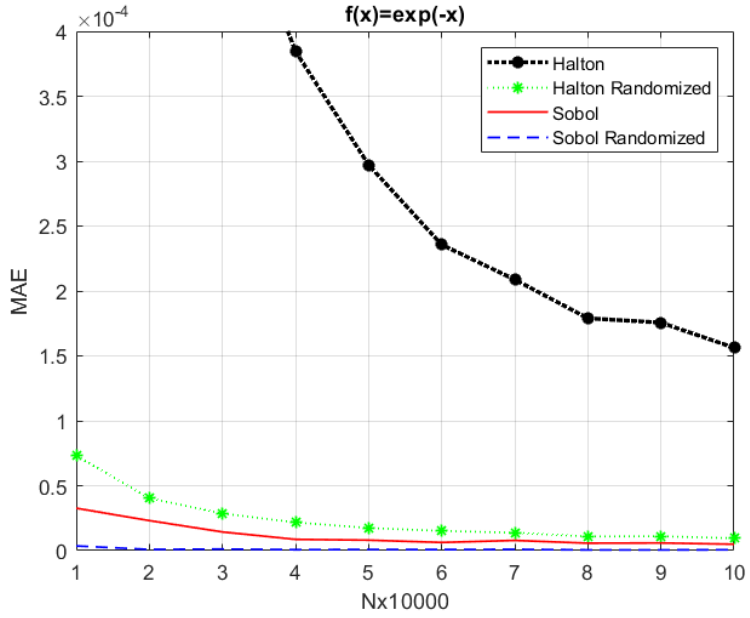


Figure 7.3: MAE as a function of sample size used, note that x-axis is to be multiplied with the order 10^4, as shown by the legend, thus the last point is using 10^5 points to evaluate the MAE on the integrals. Dimension being used here is 60.

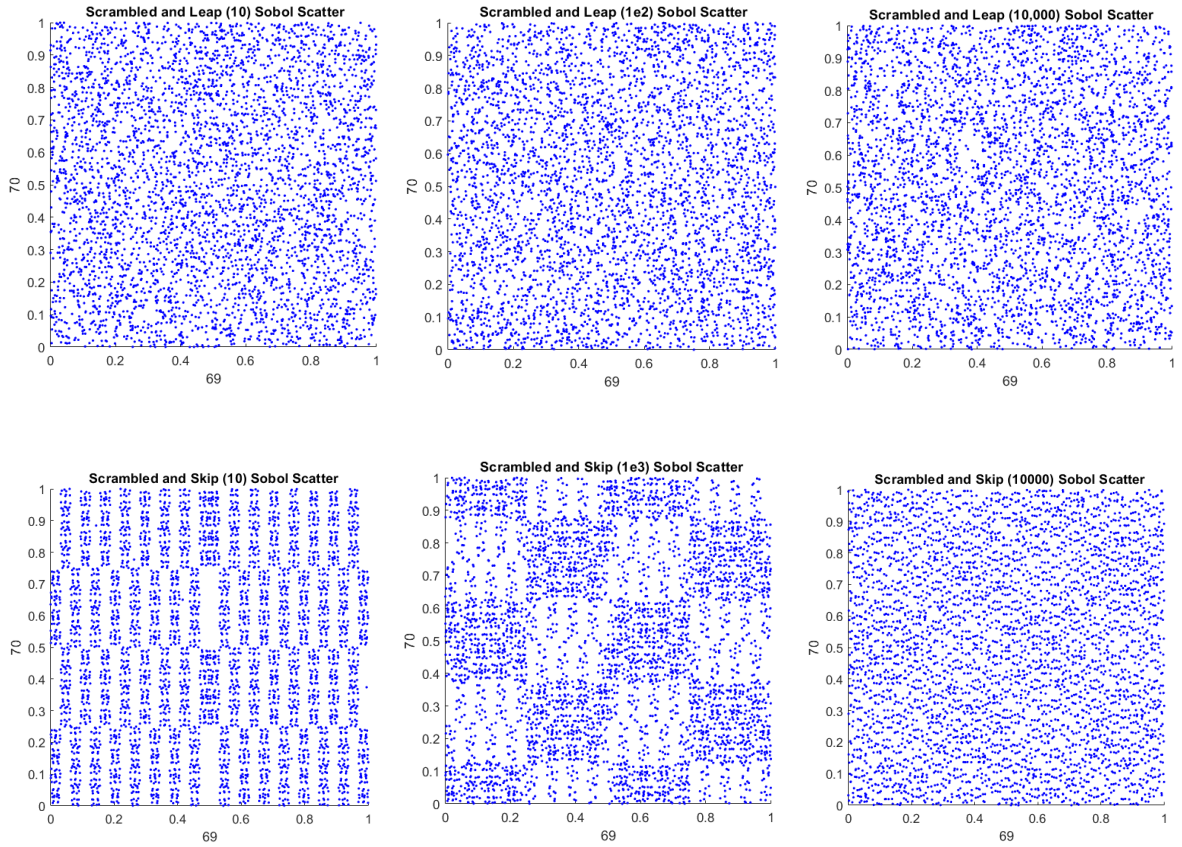
Figure (7.3) here illustrates the absolute error increase with the sample size n. We are increasing the sample size with 10.000 samples for every step on the $x-$axis. In general the reason as to why we see Halton sequence having trouble in high dimension has something to do with the discrepancy deteriorating badly for the Halton sequence.

[2]There is no reason for this specific function and it was chosen arbitrarily

Leap and Skip

Continuing on the subject of discrepancy of the low discrepancy sequences, we will focus on Sobol sequence and the options of `Leap` and `Skip`, as previously mentioned, `leap`: Specify the number of points we will leap over in the sequence and omit for every point taken, `skip`: Number of initial points in the sequence to omit from the point set. In the plot below we demonstrate the capability of these options and how we used them for our dissertation. Again we will be doing an orthogonal projection 2D image of the first 1000 points of a Sobol sequence with different combination of the leap and skip value. Refer to figure (4.2) for a vanilla projection of the Sobol points. Here are plot for different values of leap and skip parameters.

Figure 7.4



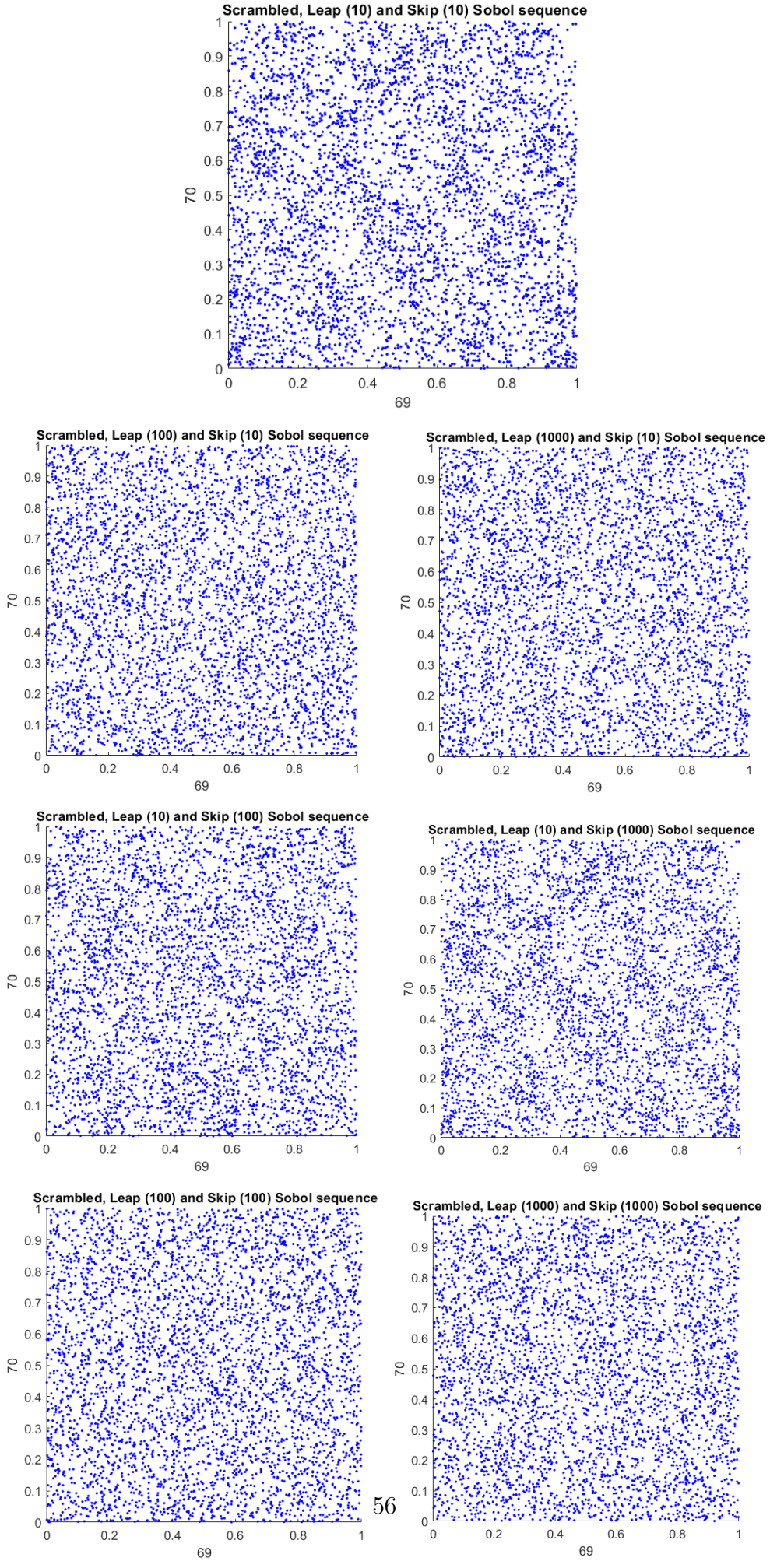
From what we see here is that a positive leaping value increases the uniform spread of the point in the the 2D projection, it is not distinguishable to see if there is a difference in choosing a leaping value being 10, 100 or 10,000. All in all we would like this value to be at least positive, as it clearly destroys the chess-like pattern alone.

A positive skip value appears also seems to decrease the size of the gaps, it is not clear from these plots either which value we would like to start out with. Therefore we will try

different combination of these skip and leap values together to see if there is something to prefer. From what is seen below it is hard to see with the human eye what is better, and traditional methods to test for randomness appears to be non-fruitions. Thus, we can only conclude that some positive value of both options is desired, and preferably higher than 10 for both as we can see some improvement visibly in the discrepancy.

it those numbers should be higher than 10 for both as it is possible to

Figure 7.5


Scrambled, Leap (10) and Skip (10) Sobol sequence
Scrambled, Leap (100) and Skip (10) Sobol sequence
Scrambled, Leap (1000) and Skip (10) Sobol sequence
Scrambled, Leap (10) and Skip (100) Sobol sequence
Scrambled, Leap (10) and Skip (1000) Sobol sequence
Scrambled, Leap (100) and Skip (100) Sobol sequence
Scrambled, Leap (1000) and Skip (1000) Sobol sequence
70
69

Appendix B

Suppose we have an SDE

$$X(t+\Delta t) = \mu(t, X(t))\Delta t + \sigma(t, X(t))dZ(t), \tag{7.1}$$

Here $Z(t)$ is a normally distributed variable independent of everything up to time t, while μ and σ are given deterministic functions. typically μ is known as the drift term and σ is called the diffusion term. To model Gaussian disturbance the introduction of Brownian motions (Wiener processes) where introduced,

Definition 17 *A stochastic process W is a called a Brownian motion if the following conditions are adhered*

1. $W(0) = 0$
2. *The process W has independent increments, i.e. if $r < s \leq t < u$ then $W(u) - W(t)$ and $W(s) - W(r)$ are independent stochastic variables.*
3. *For $s < t$ the stochastic variable $W(t) - W(s)$ has a Gaussian distribution of $N(0, \sqrt{t-s})$.*
4. *W has continuous trajectories.*

If we where to write up equation (7.1) in its integral form and driven by a Brownian motion, it would look as follows

$$X_t = \int_0^t \mu_s ds + \int_0^t \sigma_s dW_s$$

Where it is known that $\mathbb{E}[X_t] = \int_0^t \mu_s ds$ and $\text{Var}[X_t] = \int_0^t \sigma_s^2 ds$

Theorem 18 (Itô's theorem) Let a process Y_t be a stochastic process given by the following SDE

$$dY_t = \mu(t)dt + \sigma(t)dW_t$$

where let μ and σ are adapted processes and f be a $C^{1,2}$[3]. We then define X by $X_t = g(t, Y_t)$, then X has the stochastic differential given by

$$dg(t, Y_t) = \left(\frac{\partial g}{\partial t}(t, Y_t) + \mu(t)\frac{\partial g}{\partial y}(t, Y_t) + \frac{1}{2}\sigma^2(t)\frac{\partial g}{\partial y^2}(t, Y_t)\right)dt + \sigma(t)\frac{\partial g}{\partial y}(t, Y_t)dW_t$$

[3] i.e. g is twice continuously differentiable function

Remark that $[dW]^2 = dt$

Proof: Proof omitted, see Bjork, 2004. ■

Definition 19 (Geometric Brownian Motion) A process S is said to follow Geometric Brownian motion (GBM) if,

$$dS_t = \mu S_t dt + \sigma S_t dW_t, \tag{7.2}$$

where the volatility term σ and drift term μ are constant and W is a Brownian motion. Investigating the process Z, defined $Z_t = \log(S_t)$, where it is assume that S is a solution and S is strictly positive. Thus the Itô's formula gives us

$$\begin{aligned} dZ =& f'(S_t)dS_t + \frac{1}{2}f''(S_t)(dW_t)^2, \\ =& \frac{1}{S_t}\left(\mu S_t dt + \sigma S_t dW_t\right) + \frac{1}{2}(-S_t^{-2})(S_t^2\sigma^2)dt, \\ =& \left(\mu - \frac{\sigma^2}{2}\right)dt + \sigma dW_t. \end{aligned}$$

This gives us the following equation

$$dZ_t = \left(\mu - \frac{\sigma^2}{2}\right)t + \sigma W_t$$

$$Z_0 = \log(S_0)$$

Thus we can simply integrate this directly into Z; since the right-hand side contains no Z terms,

$$Z_t = \log(S_0) + \left(\mu - \frac{\sigma^2}{2}\right)t + \sigma W_t$$

This leads us to the solution of S which is given by

$$S_t = S_0 e^{\left(\mu - \frac{\sigma^2}{2}\right)t + \sigma W_t} \tag{7.3}$$

This is known as the solution to a GBM, which corresponds to most implementation of schemes and methods used in financial option pricing. A further in depth explanation on Itô and GBM

7.3.1 $(\mathcal{M}, \mu)$-Uniform Sets

Consider er the general probability space $(\Omega, \mathcal{B}, \mu)$, Ω is the arbitrary non-empty set, $\mathcal{B}$ is the σ-algebra of subsets of Ω, and lastly μ is our measure defined on $\mathcal{B}$. Then construct

a counting function $A(M : \mathcal{P})$, where $M \subseteq \Omega$ and the points set $\mathcal{P} = \{\omega_1, ..., \omega_N\}$ of the elements in Ω as

$$A(M : \mathcal{P}) = \sum_{n=1}^{N} \mathbb{1}_M(\omega_n) \tag{7.4}$$

Here $\mathbb{1}_M$ is the indicator function of M. We will define a point set as $(\mathcal{M}, \mu)$-uniform if the proportion of points that lands in the subset of $\mathcal{M}$ are equivalent to the volume of subsets, as seen in Niederreiter, 2003.

Definition 20 *Let $(\Omega, \mathcal{B}, \mu)$, Ω be the arbitrary probability space and let $\mathcal{M}$ be a non-empty subset of $\mathcal{B}$. A will be the point set of $\mathcal{P}$ containing N entries of Ω is called $(\mathcal{M}, \mu)-$uniform if*

$$A(M : \mathcal{P}) = \mu(M)N$$

for all $M \in \mathcal{M}$.

In Niederreiter, 2003 gives a rather nice example on the usage of the construction $(\mathcal{M}, \mu)$-uniform in relation to the (t, m, d)-net[4]. The example shows how the (t, m, d)-net together with the uniform point set property make it attractive for quasi-Monte Carlo integration. In the following theorem, Niederreiter provided the general foundation to derive the error bounds under his new definition $(\mathcal{M}, \mu)$-uniform sets. Thus getting rid of the assumption of bounded variation of the integrand, and consequently freeing the error bound of the same.

[4]See example 2 in **page 286**; Niederreiter, 2003

Appendix C

Pseudo-algorithms for figure (7.2)

Algorithm 1 Program 1: Quasi-Random Sequence's Absolute Error

$N \leftarrow$ define number of samples
$d \leftarrow$ define the size of dimension
$x \leftarrow$ calculate the exact value of the integral
$s \leftarrow$Use `haltonset`, `sobolset` to generate $N \times d$ matrix
Use `scramble` on s with corresponding scrambling method for randomized sequence
for $j = 1$ to d **do**
    for $i = 1$ to N **do**
        $y[i] = f(s_i)$
    Integral$_{est} = (b - a) \cdot \text{mean}(y)$ Error$[j] = |\text{Integral}_{est} - x|$

Pseudo-algorithms for figure (7.3)

Algorithm 2 Program 2: Convergence in Absolute Error of Quasi-Random sequences

$N \leftarrow$ define number of samples
$d \leftarrow$ define the size of dimension
$x \leftarrow$ calculate the exact value of the integral
$k \leftarrow 1$
Use `scramble` on s with corresponding scrambling method for randomized sequence
for $N = 10.000$ to 100.000 stepsize 10.000 **do**
    $s \leftarrow$Use `haltonset`, `sobolset` to generate $N \times d$ matrix
    for $j = 1$ to d **do**
        for $i = 1$ to N **do**
            $y[i] = f(s_i)$
        $yy[j] =$mean(y)
    Integral$_{est} = (b - a) \cdot \text{mean}(yy)$ Error$[k] = |\text{Integral}_{est} - x|$ $k = k + 1$

Bibliography

Krause, M. (1903). Fouriersche reihen mit zwei veränderlichen grössen. *Ber. Sächs. Akad. Wiss. Leipzig*, *55*, 164–197.

Hardy, G. H. (1906). On double fourier series and especially those which represent the double zeta-function with real and incommensurable parameters. *Quart. J. Math*, *37*(1), 53–79.

Koksma, J. F. (1942). A general theorem from the theory of uniform distribution modulo 1. *Mathematica B (Zutphen)*, *11*, 7–11.

Hlawka, E. (1961). Funktionen von beschränkter variatiou in der theorie der gleichverteilung. *Annali di Matematica Pura ed Applicata*, *54*(1), 325–333.

Sobol, I. M. (1967). On the distribution of points in a cube and the approximate evaluation of integrals. *Zhurnal Vychislitel'noi Matematiki i Matematicheskoi Fiziki*, *7*(4), 784–802.

Antonov, I. A., & Saleev, V. (1979). An economic method of computing lpτ-sequences. *USSR Computational Mathematics and Mathematical Physics*, *19*(1), 252–256.

Pitman, J., & Yor, M. (1982). A decomposition of bessel bridges. *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, *59*(4), 425–457.

Fox, B. L. (1986). Algorithm 647: Implementation and relative efficiency of quasirandom sequence generators. *ACM Transactions on Mathematical Software (TOMS)*, *12*(4), 362–376.

Bratley, P., & Fox, B. L. (1988). Algorithm 659: Implementing sobol's quasirandom sequence generator. *ACM Transactions on Mathematical Software (TOMS)*, *14*(1), 88–100.

Niederreiter, H. (1992). *Random number generation and quasi-monte carlo methods*. SIAM.

Morokoff, W. J., & Caflisch, R. E. (1994). Quasi-random sequences and their discrepancies. *SIAM Journal on Scientific Computing*, *15*(6), 1251–1279.

Birge, J. R. (1995). *Quasi-monte carlo approaches to option pricing* (tech. rep.).

Owen, A. B. (1995). Randomly permuted (t, m, s)-nets and (t, s)-sequences. *Monte carlo and quasi-monte carlo methods in scientific computing* (pp. 299–317). Springer.

Paskov, S., & Traub, J. F. (1996). Faster valuation of financial derivatives.

Boyle, P., Broadie, M., & Glasserman, P. (1997). Monte carlo methods for security pricing. *Journal of economic dynamics and control*, *21*(8-9), 1267–1321.

Kocis, L., & Whiten, W. J. (1997). Computational investigations of low-discrepancy sequences. *ACM Transactions on Mathematical Software (TOMS)*, *23*(2), 266–294.

Paskov, S. (1997). New methodologies for valuing derivatives, in "mathematics of derivative securities," s. pliska and m. dempster eds., isaac newton institute.

Caflisch, R. E. (1998). Monte carlo and quasi-monte carlo methods. *Acta numerica*, *7*, 1–49.

Matoušek, J. (1998). On thel2-discrepancy for anchored boxes. *Journal of Complexity*, *14*(4), 527–556.

Sloan, I. H., & Woźniakowski, H. (1998). When are quasi-monte carlo algorithms efficient for high dimensional integrals? *Journal of Complexity*, *14*(1), 1–33.

Carr, P., & Madan, D. (1999). Option valuation using the fast fourier transform. *Journal of computational finance*, *2*(4), 61–73.

Papageorgiou, A. (2001). Fast convergence of quasi-monte carlo for a class of isotropic integrals. *Mathematics of Computation*, *70*(233), 297–306.

Lemieux, C., Cieslak, M., & Luttmer, K. (2002). Randqmc user's guide: A package for randomized quasi-monte carlo methods in c. *Technical Rep*, *712*(15), 2002.

Hong, H. S., & Hickernell, F. J. (2003). Algorithm 823: Implementing scrambled digital sequences. *ACM Transactions on Mathematical Software (TOMS)*, *29*(2), 95–109.

Joe, S., & Kuo, F. Y. (2003). Remark on algorithm 659: Implementing sobol's quasirandom sequence generator. *ACM Transactions on Mathematical Software (TOMS)*, *29*(1), 49–57.

Niederreiter, H. (2003). Error bounds for quasi-monte carlo integration with uniform point sets. *Journal of computational and applied mathematics*, *150*(2), 283–292.

Papageorgiou, A. (2003). Sufficient conditions for fast quasi-monte carlo convergence. *Journal of Complexity*, *19*(3), 332–351.

Bjork, T. (2004). Arbitrage pricing. *Arbitrage theory in continuous time* (pp. 1–228). Oxford University Press.

Glasserman, P. (2004). *Monte carlo methods in financial engineering* (Vol. 53). Springer.

Jiang, J. X., & Birge, J. R. (2004). Error bounds for quasi-monte carlo methods in option pricing. *Available at SSRN 634161*.

Broadie, M., & Kaya, Ö. (2006). Exact simulation of stochastic volatility and other affine jump diffusion processes. *Operations research*, *54*(2), 217–231.

Andersen, L. B. (2007). Efficient simulation of the heston stochastic volatility model. *Available at SSRN 946405*.

Joe, S., & Kuo, F. Y. (2008). Notes on generating sobol sequences. *ACM Transactions on Mathematical Software (TOMS)*, *29*(1), 49–57.

Lord, R., Koekkoek, R., & Dijk, D. V. (2010). A comparison of biased simulation schemes for stochastic volatility models. *Quantitative Finance*, *10*(2), 177–194.

Ökten, G., & Willyard, M. (2010). Parameterization based on randomized quasi-monte carlo methods. *Parallel Computing*, *36*(7), 415–422.

Gatheral, J. (2011). *The volatility surface: A practitioner's guide* (Vol. 357). John Wiley & Sons.

Baldeaux, J., & Roberts, D. (2012). Quasi-monte carlo methods for the heston model. *arXiv preprint arXiv:1202.3217*.

Faure, H., & Kritzer, P. (2013). New star discrepancy bounds for (t, m, s)-nets and (t, s)-sequences. *Monatshefte für Mathematik*, *172*(1), 55–75.

Kienitz, J., & Wetterau, D. (2013). *Financial modelling: Theory, implementation and practice with matlab source*. John Wiley & Sons.

L'Ecuyer, P. (2014). Randomized quasi-monte carlo. *une section de Wiley StatsRef: Statistics Reference Online, Article*, *8240*.

Siegenthaler, W. (2020). *Sobol*. Retrieved February 24, 2022, from %5Curl%7Bhttps://github.com/Wsiegenthaler/sobol-rs%7D