Option Pricing

Option Pricing - Multi-Level Monte Carlo Method

Georgii NIKISHIN, Julien SKLARIK, and Tristan KIRSCHER

ENSAE

April 2023

1. Problem statement
2. MLMC
3. QMC
4. Brownian Bridge
5. Comparison Results

1 Problem statement
Option Pricing
CIR Model

2 MLMC

3 QMC

4 Brownian Bridge

5 Comparison Results

1 Problem statement
Option Pricing
CIR Model

2 MLMC

3 QMC

4 Brownian Bridge

5 Comparison Results

Problem statement

We wish to evaluate the price of an Asian call option:

$$C = \mathbb{E}\left[e^{-rT}\left(\frac{1}{k}\sum_{i=1}^{k} S(t_i) - K\right)^+\right] \tag{1}$$

Problem statement

The previous equation comes from the following option prices:

Options prices

The value of an Asian call option, denoted by $Call_{Asian}$, is equal to the maximum of zero and the difference between the average price of the underlying asset, $\frac{1}{n}\sum_{i=1}^{n} S_i$, and the strike price, K.

$$Call_{Asian} = max\left(\frac{1}{n}\sum_{i=1}^{n} S_i - K, 0\right) \tag{2}$$

$$Put_{Asian} = max\left(K - \frac{1}{n}\sum_{i=1}^{n} S_i, 0\right) \tag{3}$$

1 Problem statement
Option Pricing
CIR Model

2 MLMC

3 QMC

4 Brownian Bridge

5 Comparison Results

CIR Model

To simulate the stock path we used the CIR (Cox, Ingersoll, Ross) model:

Stock path

$$dS_t = \alpha(\beta - S_t)dt + \sigma\sqrt{S_t}dW_t \tag{4}$$

where W_t is a Brownian motion.

The parameter α corresponds to the speed of adjustment to the mean β, and σ to volatility. $\alpha(\beta - S_t)$ is the "drift factor".
We took:
$\alpha = 0.15, \beta = 0.2, \sigma = 0.2$, $T = 1$, $r = 0.05$, $K = 4$, $k = 20$, $t_i = i/20$.

CIR Model

Recall the previous CIR Model equation :
$dS_t = \alpha(\beta - S_t)dt + \sigma\sqrt{S_t}dW_t$

Euler discretization (cf. slide 87 from course)

$$S_{t+1} = S_t + \alpha(b - S_t)\Delta t + \sigma\epsilon\sqrt{S_t} \tag{5}$$

- Δt : discretization step
- $\epsilon \sim \mathcal{N}(0, \Delta t)$

Choice of Δt : trade-off between discretization bias and CPU time

1. Problem statement
2. MLMC
   - Overview
   - Algorithm
   - Variance
   - Algorithm
3. QMC
4. Brownian Bridge
5. Comparison Results

1 Problem statement

2 MLMC
- Overview
- Algorithm
- Variance
- Algorithm

3 QMC

4 Brownian Bridge

5 Comparison Results

Overview

- The multi-level Monte Carlo (MLMC) method is a variance reduction technique for estimating expectations of expensive-to-evaluate functions (here, the price of an Asian call option).
- The MLMC method can be applied to a wide range of problems, including SDEs and option pricing.
- The method uses multiple levels of approximations to reduce the computational cost of the estimation.

1 Problem statement

2 MLMC
Overview
Algorithm
Variance
Algorithm

3 QMC

4 Brownian Bridge

5 Comparison Results

The multi-level Monte Carlo method (cf. Mike Giles (Oxford))

Two-level Monte Carlo

If we want to estimate $\mathbb{E}[\widehat{P}_1]$ but it is much cheaper to simulate $\widehat{P}_0 \approx \widehat{P}_1$, then since

$$\mathbb{E}[\widehat{P}_1] = \mathbb{E}[\widehat{P}_0] + \mathbb{E}[\widehat{P}_1 - \widehat{P}_0]$$

we can use the estimator

$$N_0^{-1} \sum_{n=1}^{N_0} \widehat{P}_0^{(0,n)} \;+\; N_1^{-1} \sum_{n=1}^{N_1} \left(\widehat{P}_1^{(1,n)} - \widehat{P}_0^{(1,n)} \right)$$

Benefit: if $\widehat{P}_1 - \widehat{P}_0$ is small, its variance will be small, so won't need many samples to accurately estimate $\mathbb{E}[\widehat{P}_1 - \widehat{P}_0]$, so cost will be reduced greatly.

The multi-level Monte Carlo method (cf. slide 88 from the course)

Multilevel Monte Carlo

Natural generalisation: given a sequence $\widehat{P}_0, \widehat{P}_1, \ldots, \widehat{P}_L$

$$\mathbb{E}[\widehat{P}_L] = \mathbb{E}[\widehat{P}_0] + \sum_{\ell=1}^{L} \mathbb{E}[\widehat{P}_\ell - \widehat{P}_{\ell-1}]$$

we can use the estimator

$$N_0^{-1} \sum_{n=1}^{N_0} \widehat{P}_0^{(0,n)} + \sum_{\ell=1}^{L} \left\{ N_\ell^{-1} \sum_{n=1}^{N_\ell} \left(\widehat{P}_\ell^{(\ell,n)} - \widehat{P}_{\ell-1}^{(\ell,n)} \right) \right\}$$

with independent estimation for each level of correction

The multi-level Monte Carlo method

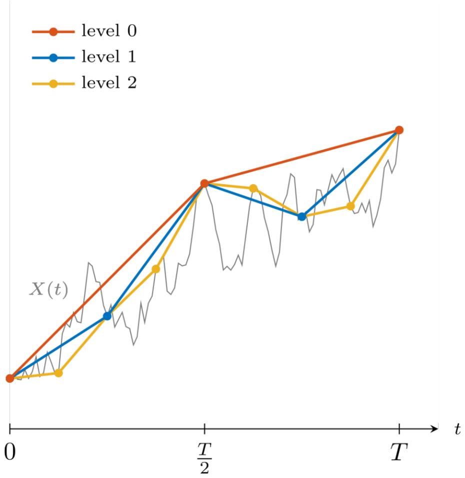


Figure 2: Approximation of a sample path of an SDE on different levels

1 Problem statement

2 MLMC
- Overview
- Algorithm
- Variance
- Algorithm

3 QMC

4 Brownian Bridge

5 Comparison Results

The multi-level Monte Carlo method

Variance

The variance of this simple estimator is $\mathbb{V}[\hat{Y}_l] = N_l^{-1} V_l$ where V_l is the variance of a single sample.
The variance of the combined estimator $\hat{Y} = \sum_{l=0}^{L} \hat{Y}_l$:

$$\mathbb{V}[\hat{Y}] = \sum_{l=0}^{L} N_l^{-1} V_l$$

The computational cost, if one ignores the asymptotically negligible cost of the final payoff evaluation, is proportional to:

$$\sum N_l h_l^{-1}$$

where h_l is a step size at level l

1. Problem statement
2. MLMC
   - Overview
   - Algorithm
   - Variance
   - Algorithm
3. QMC
4. Brownian Bridge
5. Comparison Results

The multi-level Monte Carlo method

Numerical algorithm

1. start with L=0
2. estimate V_L using an initial $N_L = 10^4$ samples
3. define optimal $N_l, l = 0, \dots, L$ using Eqn. (6)
4. evaluate extra samples at each level as needed for new N_l
5. if $L \geq 2$, test for convergence using Eqn. (7)
6. if $L < 2$ or not converged, set $L := L + 1$ and go to 2.

$$N_l = 2\varepsilon^{-2}\sqrt{V_l h_l}\left(\sqrt{V_l/h_l}\right) \tag{6}$$

$$|\hat{Y}_L - M^{-1}\hat{Y}_{L-1}| < \frac{1}{\sqrt{2}}(M^2 - 1)\varepsilon. \tag{7}$$

CPU Time and Discretisation parameter comparison

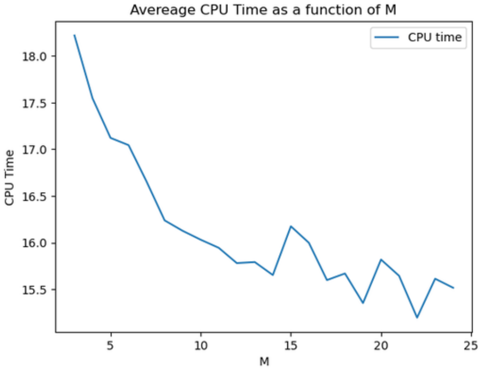


Figure 3: Plot of the CPU Time average as a function of M for the MLMC method

1. Problem statement

2. MLMC

3. QMC
   - Idea behind QMC
   - Randomized QMC

4. Brownian Bridge

5. Comparison Results

1 Problem statement

2 MLMC

3 QMC
   Idea behind QMC
   Randomized QMC

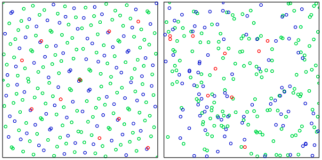
4 Brownian Bridge

5 Comparison Results

Idea behind QMC

- The idea behind QMC is to use a **deterministic sequence of points** that are more evenly spaced throughout the integration region, leading to faster convergence and more accurate estimates of integrals compared to traditional MC methods.
- The points are typically generated using **low-discrepancy sequences** such as *Sobol* sequences, *Halton* sequences, or *Faure* sequences.

Low discrepancy sequence



256 points from the first 256 points for the 2,3 *Sobol* sequence (left) compared with a pseudorandom number source (right). The *Sobol* sequence covers the space more evenly.

Source: Sobol, I.M. (1967), "Distribution of points in a cube and approximate evaluation of integrals". Zh. Vych. Mat. Mat. Fiz.

1. Problem statement
2. MLMC
3. QMC
   - Idea behind QMC
   - Randomized QMC
4. Brownian Bridge
5. Comparison Results

Randomized QMC

Randomization allows to give an estimate of the variance while still using quasi-random sequences :
Let $x_1, ..., x_N$ be the point set from the low discrepancy sequence.
We sample s-dimensional random vector U and mix it with $x_1, ..., x_N$:

- for each x_j, create $y_j = x_j + U \pmod 1$
- use sequence (j_y) instead of (x_j)

Compared to standard Monte-Carlo, the variance and the computation speed are slightly better from the experimental results in Tuffin (2008)

Option prices output

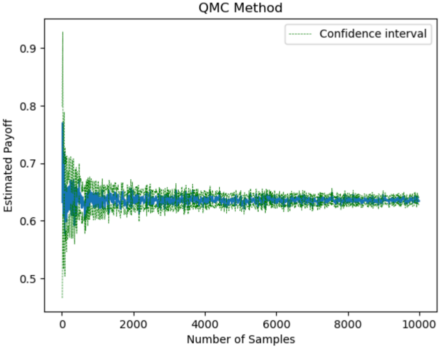


Figure 4: Plot of option prices using the quasi-Monte Carlo method with the Sobol sequence

CPU Time and Variance comparison

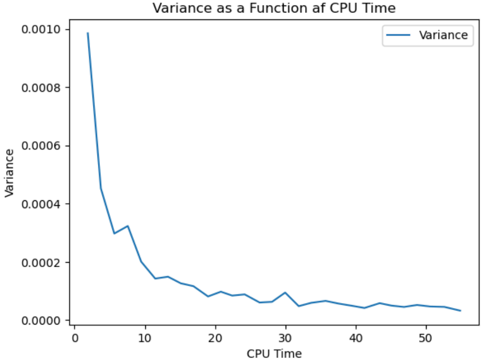


Figure 5: Plot of the Variance as a function of CPU Time for the Quasi-MC method

1 Problem statement

2 MLMC

3 QMC

4 Brownian Bridge

5 Comparison Results

Simulating Brownian paths

Random walk: $W_{t_i} | W_{t_{i-1}} \sim \mathcal{N}(W_{t_{i-1}}, t_i - t_{i-1})$

Brownian Bridge: $B_{t_i} = W_{t_i} | W_{t_{i-1}}$

$$W_{t_{i+1}} | W_{t_{i-1}} \sim \mathcal{N}\left(\frac{(t_{i+1}-t_i)W_{t_{i-1}} + (t_i - t_{i-1})W_{t_{i+1}}}{t_{i+1}-t_{i-1}}, \frac{(t_{i+1}-t_i)(t_i - t_{i-1})}{t_{i+1}-t_{i-1}}\right)$$

But the increments, B_{t_i}, are not independent.

Let $t \in (t_1, t_2)$, $B(t_1) = \mathrm{a}$ and $B(t_2) = b$.

$$B \sim \mathcal{N}\left(a + \frac{t - t_1}{t_2 - t_1}(b - a), \frac{(t_2 - t)(t - t_1)}{t_2 - t_1}\right)$$

Paths visualisation

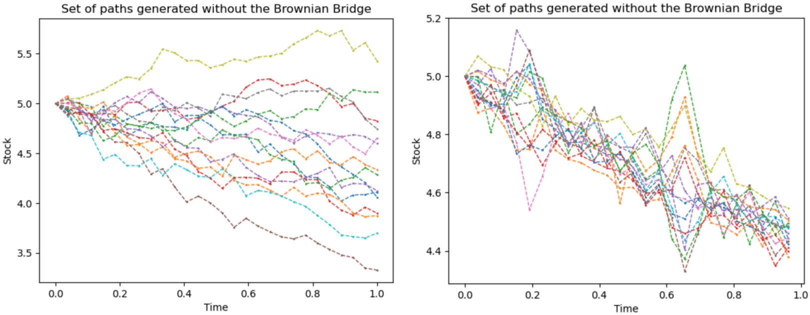


Figure 6: Set of generated paths

CPU Time and Discretisation parameter comparison

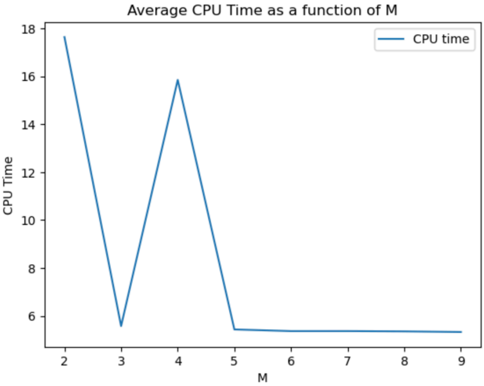


Figure 7: Plot of the CPU Time average as a function of M for the MLMC using the Brownian Bridge

1 Problem statement

2 MLMC

3 QMC

4 Brownian Bridge

5 Comparison Results

Main Results

- CPU times are given for a fixed Variance of 2.1×10^{-5}
- Variance values are given for a fixed CPU Time of 7.0s

	MC	MLMC	QMC	Bridge MC
CPU Time	21.9s	7.0s	$> 83.0s$	4.6s
Variance	7.5×10^{-5}	2.1×10^{-5}	2.6×10^{-4}	3.6×10^{-6}

Results for Brownian Bridge MLMC

- CPU times are given for a fixed Variance of 1.7×10^{-5}
- Variance values are given for a fixed CPU Time of 5.9s

	Bridge MC	Bridge MLMC
CPU Time	4.4s	5.9s
Variance	1.2×10^{-5}	1.7×10^{-5}

Thank you for your attention!