Computational Finance Project 4

Kai Chen, Rutgers University 03/06/2018

Introduction

This paper reports the main work of Project 4 for Computational Finance (Math 16:642:623). In this project, Sobol sequence is implemented to generate Quasi-random numbers for option pricing by Monte Carlo simulation. The results are compared with the prices given by Park-Miller Pseudo-random number generator.

Preliminary

The option used in this project is a European-style vanilla call. We assume that the stock price is geometric Brownian motion and all the needed variables are constant:

- Volatility $\sigma = 0.3$
- Initial asset price $S(0) = 100$
- Risk-free interest rate $r = 0.05$
- Strike $K = 110$
- Maturity $T = 1$ year

Closed-form Formula

The arbitrage-free call option price implied by Black-Scholes-Merton model is:

$$C(S_t, t) = e^{-r(T-t)}[S_t e^{r(T-t)} N(d_1) - KN(d_2)]$$

where,

$$d_1 = \frac{1}{\sigma\sqrt{T-t}}[\ln(\frac{S_t}{K}) + (r + \frac{1}{2}\sigma^2)(T-t)]$$

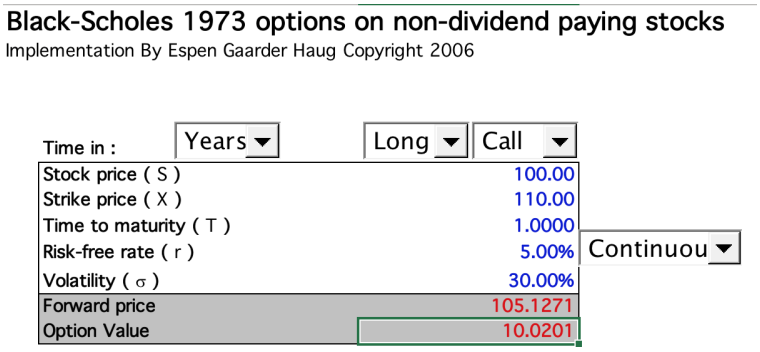
$$d_2 = d_1 - \sigma\sqrt{T-t}$$

The prices of this European call option calculated by our program is:

$$c = 10.0201$$

Benchmark

This result is identical to the benchmark that we use, as is shown in the following screen-shots:



Random Number Generation

Pseudo-Random Number

Random number generation is an important step in Monte Carlo simulation. However, it is not easy to make a human made programmed computer to give a real random number sequence that people can observe in physical phenomenon. What we can do is to design some algorithms to let computers generate some so called "Pseudo-random numbers", which typically exhibit statistical randomness while being generated by an entirely deterministic causal process. So the entire sequence of numbers is only as powerful as the randomly chosen parts - sometimes the algorithm and the seed, but usually only the seed.

Although these kinds of random processes are not completely random, they still perform very well and widely being used in Monte Carlo simulation. Here, we use Park-Miller to do the simulation and price the call option. "Variance Reduction" technique (Antithetic Method), which is introduced in Project 2, is turned on and off respectively. The results are given as:

$$c(ParkMiller) = 10.2803$$

$$c(ParkMiller\&Antithetics) = 10.1288$$

Quasi-Random Number

The main drawback of Pseudo-random generators is that once given the seed, the random sequence will become completely deterministic. In addition, Pseudo-random numbers might not be evenly distributed enough.

In order to solve these problems, Quasi-random generator is invented. The "quasi" modifier is used to denote more clearly that the values of a low-discrepancy sequence are neither random

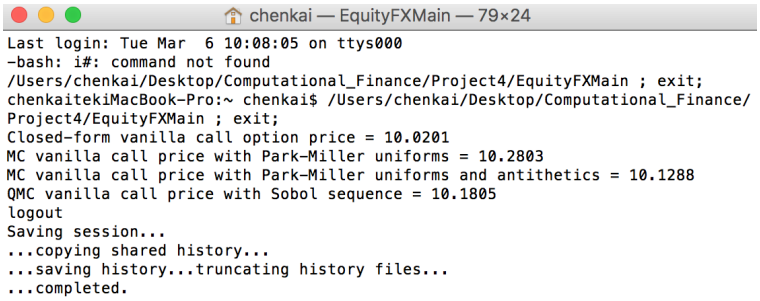
nor Pseudo-random, but such sequences share some properties of random variables and in certain applications such as the quasi-Monte Carlo method their lower discrepancy is an important advantage.

In this project, we use Sobol sequence for Monte Carlo simulation. The generator is coded by J.Burkhardt and it is adapted by us so that it could be incorporated into Joshi's EquityFXMain pricing structure. One thing needs to be mentioned is that "variance reduction" is not appropriate for Quasi-random sequences. So we turn off Antithetic Method when pricing with Sobol sequence. The result is given as:

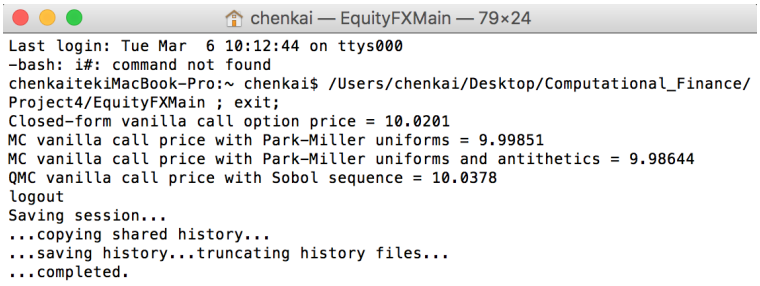
$$c(Sobol) = 10.1805$$

Result Analysis

The results given by our program are summarized in the following screen-shot:



In order to verify our program, we test the convergence of all the methods by increasing the simulation paths. Here we list the result when using 100,000 paths:



As we can see, the results given by all the methods are converging to the closed-form solution. Also notice that Sobol sequence generator gives better estimation on average, which shows the prominence of Quasi-Monte Carlo methods.

Questions

- In Sobol sequence generating process, we cannot use Marsaglia-Bray-Box-Muller algorithm. The reason is that Box-Muller algorithm is a pseudo-random number sampling method for generating pairs of independent, standard, normally distributed random numbers. However, in Sobol sequence, in order to form successively finer uniform partitions of the unit interval and then reorder the coordinates in each dimension, the numbers in the sequence are not exactly independent, which is also the reason we call it "Quasi". Thus, inverse transformation algorithm is a better choice over Box-Muller since it does not have that independent requirement.
- The generated pairs are included in the submitted files. Park-Miller pairs are generated by calling the function twice for each step. However, the Sobol sequence is set up as 2-dimensional in order to have closer approximation. We can notice that Sobol pairs have low discrepancy as expected. The graphs are shown as follows: