Simulation Methods in Financial Mathematics
Computer Lab 12

Goals of the lab:

- To learn the generation of Halton and Sobol sequences for quasi-Monte Carlo simulation
- To learn to compute the price of an European option by using quasi-Monte Carlo method

The speed of convergence of Monte-Carlo methods for computing expected values is always $\frac{c}{\sqrt{n}}$, where n is the number of random variables generated. It is possible to reduce the constant c, but for Monte-Carlo methods the convergence speed with respect to n is always the same. It turns out that if we replace the random numbers in a Monte-Carlo method by specially constructed (non-random) values, it is possible to improve the convergence rate to $\frac{C\log(n)^d}{n}$, where d is the dimension of the points. Since $\log(n)$ increases very slowly as n increases, the gain in accuracy is almost proportional to the number of values generated.

The basis for this improvement are so called low discrepancy sequences which are sequences in the m-dimensional unit cube $[0,1]^m$ which for every value of n cover the cube in some sense as well as possible. More precisely, a sequence of d-dimensional points $x_1, x_2, \ldots$ is called a low discrepancy sequence if there exists a constant c such that for every n and every box B of the form $[a_1,b_1]\times[a_2,b_2]\times\ldots\times[a_d,b_d]$, where $0 \le a_i < b_i \le 1$, we have

$$\left|\frac{|\{i\,:\,i\le n,\,x_i\in B\}|}{n} - Vol(B)\right| \le c\frac{log(n)^d}{n}.$$

Here the absolute value of a set denotes the number of elements in the set. In words, this inequality tells us that the proportion of the points $x_1,\ldots,x_n$ that fall into a given box B is quite close to the volume of the box for every box, so the sequence covers the unit cube well. It turns out that for such sequences (under some additional technical assumptions about the function f, see Koskma-Hlawka inequality) we have

$$\left|E\,f(X) - \frac{1}{n}\sum_{i=1}^{n} f(x_i)\right| \le C_f\frac{log(n)^d}{n},$$

X is a d-dimensional vector of independent uniformly distributed random variables. As we'll see, the condition that X has to be a uniformly distributed random vector, is not a problem - if we have a Monte Carlo method that uses random variables from some other distribution, we can use the idea of applying the inverse cumulative distribution function to change it so that uniformly distributed random variables are used.

One (the simplest possible) class of such sequences is so called Halton sequences.

The idea of Halton sequences is as follows. We can represent a number, say k, in a number system that has base b where $b \ge 2$, in the form

$$k = \sum_{i=0}^{\infty}\alpha_i(k)b^i,\ \ \alpha_i(k)\in\{0,1,\ldots,b-1\}.$$

Denote

$$\psi_b(k) = \sum_{i=0}^{\infty}\frac{\alpha_i(k)}{b^{i+1}}.$$

Then it turns out that the sequence $\psi_b(0), \psi_b(1), \ldots$ is a one-dimensional low discrepancy sequence. When generating an m-dimensional Halton sequence, m numbers $b_1, b_2, \ldots, b_m$ that do not have a common factor (usually m first prime numbers) are chosen and point $\mathbf{x}_k$ is defined as a point with coordinates

$$x_{kj} = \psi_{b_j}(k),\ \ j = 1,\ldots,m.$$

Task 1 Install and load package `randtoolbox`. Installation can usually be done by the command `install.packages("randtoolbox")` and loading by the command `library("randtoolbox")`. If you do not have administrative rights and R is not configured well, you still can install packages for your own use. For this make a folder where you are going to install your own R packages. Then you can use commands

```
.libPaths("your folder paths")
install.packages("randtoolbox",lib="your folder path")
```

to install the package. Use the function `halton(n,dim)` to generate 200 Halton points with dimension 8. Visualize the placement by scatterplots that can be created with R function `pairs()`.

There are many different low discrepancy sequences available, we'll consider Sobol sequences in addition to Halton sequences. When low discrepancy sequence is used for computing approximately an expected value of a random variable, the method is called Quasi Monte Carlo method.

The procedure of implementing a Quasi Monte Carlo methods is as follows:

1. Implement a Monte-Carlo method for for computing EY, where Y is the random variable we are interested in. It can be a simple MC or a method using various variance reduction techniques.
2. Rewrite your MC method so that it is based on generating independent uniformly distributed random variables. This can be achieved, for example, by applying the knowledge that if F_X is the cumulative distribution function of a continuous random variable X, then the random variable $F_X^{-1}(U)$, where $U \sim U(0,1)$ is uniformly distributed in the interval $(0,1)$, has the same distribution as X. So, if our MC method uses random variables from the distribution $N(0,a)$, then we can generate them by
$$X = a\Phi^{-1}(U),$$
where $U \sim U(0,1)$ and Φ is the cumulative distribution function of the standard normal distribution.
3. If for generating one value of Y we use m uniformly distributed random variables, then replace them by coordinates of a point of a m-dimensional low discrepancy sequence.
4. Compute the final answer using as the simple average of the largest number of values of Y that can be generated in reasonable time.

The final answer is usually much more accurate than the answer obtained with a simple MC using the same number of generated values of Y. Unfortunately there are no efficient procedures for estimating the error of the answer obtained by this procedure that is called Quasi Monte Carlo method.

Task 2 Consider pricing an European put option with $E = 100$ at time $t = 0$ using the Euler's method when $r = 0.1$, $D = 0$, $T = 0.5$, $S(0) = 105$ and $\sigma(t,s) = 0.6 - 0.5\,e^{-0.01s}$. We want to study how much the use of Halton sequence speeds up the convergence of the MC method in the cases when $m = 5$ and $m = 20$. To do this, replace the increments of the Brownian motion with values that are generated using a Halton sequence and find the errors for cases $n = 10000, 20000, 40000, 80000$. For comparison, find the errors for regular MC method for the same values of n. Use the knowledge that the expected value is $7,731$ when $m = 5$ and it is $7,577$ when $m = 20$.

Task 3 Repeat the computations of Task 2 with Sobol sequences.