Simulation Methods in Financial Mathematics
Computer Lab 1

Goals of the lab:

- To get familiar with R and RStudio
- To learn to use the Monte-Carlo method and estimate its error
- To learn to calculate integrals using simulation methods

Monte-Carlo method or simulation method is a computational algorithm for estimating the mean or some other characteristic of a random variable. In this course we reduce problems in finance to estimating the mean of an appropriate random variable. When estimating the mean value, MC method is based on performing independent trials and averaging the obtained results. That is, if we are interested in $\mathbb{E}Y$ for some random variable Y, we generate n independent values $Y_1, Y_2, \ldots, Y_n$ from the distribution of Y and estimate

$$\mathbb{E}Y \approx H_n = \frac{1}{n}\sum_{i=1}^{n} Y_i.$$

According to the Law of Large Numbers this average converges to the expected value of Y as the number of trials goes to infinity. Actually, there are different versions of this law, corresponding to different modes of convergence (do you know in what sense the convergence takes place?).

From the practical point of view it is not enough to know that we get the correct result when the number of trials goes to infinity. Instead, we need to know how large can the error of our estimate H_n be after a finite number of trials. Since the number of trials is usually very large when MC method is used (often in millions), an estimate of the error of H_n can be obtained by using the Central Limit Theorem: for large enough n we have that the distribution of the error of H_n (i.e. $|H_n - \mathbb{E}Y|$) is very close to the normal distribution $N(0, \frac{\sigma_Y}{\sqrt{n}})$ and hence the inequality

$$|H_n - \mathbb{E}Y| \le \varepsilon = -\frac{\Phi^{-1}(\alpha/2)\sigma_Y}{\sqrt{n}}$$

holds with approximate probability $1-\alpha$ (prove it!). Here Φ denotes the cumulative distribution function of the standard normal distribution. The inverse of a cumulative distribution function is called the *quantile function* (also *percent point function*) of that distribution, so actually the quantile function of the standard normal distribution is used in the error estimate. The standard deviation σ_Y of the random variable Y is also estimated by using $Y_1, \ldots, Y_n$.

In most cases relevant Y can be expressed as $Y = g(X)$, where X is a random variable (or random vector) with known distribution and g is some given function. In this case we generate values of Y by applying the function g to the generated values of X.

Tasks:

1. We start using the MC method. Let $Y = X^2$, where X has the standard uniform distribution. Using the sample of size $n = 1000$ find an estimate of $\mathbb{E}Y$, calculate the error estimate for $\alpha = 0.1$ and the actual error.

2. Let us write our first useful function for applying MC method in many different situations. Namely, define the function `MC1` with four arguments: the name of a function g, the name of a function that for a given n generates n random variables X, the number n of random variables to be generated, and the value of α used in computing the error estimate. The function should return a vector of two numbers; the estimate of $\mathbb{E}[g(X)]$ and the error estimate. Additionally, define the function $f(x) = x^2$ and compute the value $MC1(f, runif, 100, 0.1)$. Is the result correct?

3. Practice using Moodle VPL exercise by submitting your function to the VPL_exercise_1 in Moodle. Try to submit originally a code with some mistakes and then a correct one. You get warnings about security risks and should allow your computer to connect to the exercise server.

4. Use the function $MC1$ to repeat the first task 100 times and produce three vectors: average, error_estimate, actual_error. How many times did the actual error exceed the error estimate?

5. When using the MC method, we usually do not know beforehand how large a sample should be generated in order to get an answer that is accurate enough for our purposes. Thus simulation continues until the required precision is achieved (or in some cases until we cannot wait any longer). In order to do that we first set the number of random variables to be generated at one go (denote it by n_0) and after generating this number of values we estimate the error. If the error estimate is not small enough we generate additional n_0 values and estimate the error again by using all $2n_0$ generated values. If the error estimate is still too large, we generate again additional n_0 values and again estimate the error by using all generated values and so on. Since the number of generations needed for achieving the desired accuracy can be very large, we do not store the previously generated random variables (to avoid memory problems), and thus we cannot use the R functions to calculate the mean and standard deviation of all generated values. Instead we store only the sum, the sum of squares and the total number of values of Y generated so far. The standard deviation can then be estimated as

$$\sigma_Y \approx \sqrt{\frac{|sum\_of\_squares\_of\_y - (sum\_of\_y)^2/n|}{n-1}}.$$

Write a function which takes as the input a function g, a function gen which generates values from the distribution of X, allowed error ε and α (the probability of exceeding the allowed error) and returns the estimate (with given precision with probability $1-\alpha$) of the expected value of $Y = g(X)$.

6. Definite integral

$$\int_a^b g(x)\,dx$$

can be viewed as the expected value of a function by multiplying and dividing the integrand by a suitable probability density function:

$$\int_a^b g(x)\,dx = \int_a^b \frac{g(x)}{f_X(x)} f_X(x)\,ds = \mathbb{E}(\tilde{g}(X)),$$

where X is a random variable with the probability density function f_X such that $f_X(x) > 0,\ x \in [a,b]$ and

$$\tilde{g}(x) = \frac{g(x)}{f_X(x)} I_{[a,b]}(x).$$

Here $I_{[a,b]}(x)$ is the indicator function of the set $[a,b]$ having value 1 when x belongs to that interval and 0 otherwise and it is not needed if the density f_X or the function g is constantly zero outside of the interval $[a,b]$. So there are many ways to compute by MC the same definite integral, different choices of the random variable X give different methods with different convergence properties. As the error estimate of MC method depends on the variance of the random variable $Y = \tilde{g}(X)$, a good choice of the distribution for X should be such that in the region where $f_X(x)$ is large, the function $\tilde{g}(x)$ is close to a constant.

Use the Monte-Carlo method to calculate approximately with the precision $\varepsilon = 0.02$ (using $\alpha = 0.05$) the value of the integral

$$\int_1^\infty \frac{5e^{-2x}}{x^2+1}\,dx.$$

Use the exponential distribution $Exp(\lambda)$ with rate parameter $\lambda = 2$ for converting the integral to the problem of computing the expected value. Hint: the indicator function $I_{[a,b]}(x)$ can be written in R as $(x >= a) * (x <= b)$.

7. **Homework 1** (Deadline 13.09.2020) is about computing integrals with MC method. Detailed information is given in corresponding VPL exercise in Moodle.