Simulation Methods in Financial Mathematics
Computer Lab 3

Goals of the lab:

- To implement Euler's method for approximate solution of the stochastic differential equation (SDE)
$$dS(t) = S(t) \cdot (\mu \cdot dt + \sigma \cdot dB(t))$$
together with the initial condition $S(0) = S_0$.
- To study the weak and strong convergence rate of Euler's (or, more precisely, Euler-Maruyama) method experimentally.

As we saw in the first lab, the idea of Monte-Carlo (or simulation) method is very simple: we just have to generate certain random variable X, apply a function g to generated values and calculate the average of the results together with an error estimate. It turns out that the simplicity is only apparent: it is not always easy to generate efficiently the values of a random variable and it is not always obvious what random variable and function g to use for a given problem.

The prices of many financial instruments (including options) can be expressed as expected values of certain random variables and therefore it is possible to apply MC method for computing the prices. In Lab 2 we saw that when we assume the validity of the Black-Scholes market model with constant volatility, then generating future values of stock prices is easy and hence it is quite straightforward to use Monte-Carlo method to compute option prices. When the market model does not have constant parameters (e.g. σ is not constant), the explicit distribution of the stock prices at the exercise time T is not available. Thus we have to calculate approximate stock prices using the respective trajectories of the Brownian motion. In the present lab we study the easiest method for generating the approximate stock prices – Euler's method. In order to analyze the error introduced by approximate stock price generation, we start with the market model with constant μ and σ, so that we know the exact results.

Fix the terminal value T. In order to generate the values of $S(T)$ using Euler's method, we fix m – the number of steps and denote $d\Delta t = \frac{T}{m}$. Now we calculate

$$S_i = S_{i-1}(1 + \mu\,\Delta t + \sigma X_i), \quad i = 1, 2, \ldots, m,$$

where S_0 is the current stock price (that is known), S_i are approximate values of $S(i\Delta t)$ for a given trajectory of the Brownian motion and $X_i \sim N(0, \sqrt{\Delta t})$ and are independent. This is an approximation of the stochastic differential equation and should get better when the number of time steps is increased.

When using numerical methods for solving SDEs, two types of errors are distinguished. Firstly, how much does the generated stock price differ from the exact stock price (both based on the same trajectory of the Brownian motion). The rate of convergence of this error (in m) is called the strong rate of convergence. Secondly, how rapidly does the difference between $E[g(S_m)]$ and $E[g(S(T))]$ go to zero as m increases, where S_m are the approximate stock prices at time T generated by using a numerical method (e.g. by Euler's method) and g is some function of interest. This rate is called the weak rate of convergence. Both rates are measured in terms of the maximal power q for which the respective errors are less than or equal to $\frac{c}{m^q}$ for some constant c for all values of m. It can be shown that for most reasonably "nice" functions g (for example, for functions

with bounded first derivative) the weak convergence rate is at least as large as the strong convergence rate.

It is important to understand that when calculating the price of an option by replacing $S(T)$ with its approximation produced by the Euler's method and then using the Monte-Carlo method to estimate the expected value, the error of the final result consists of two components: the error of the Monte-Carlo method which we can reduce by increasing the number of simulations n and the error introduced because of the wrong distribution of the stock prices which we can reduce by decreasing step length m. If we let n increase without a bound all that is left is the second error.

Tasks:

1. Program a function for generating n stock prices based on the Euler's method `S_euler(n,S0,m,T,mu,sigma)`. Use this function to calculate the price of an European put option when $m = 40, S0 = 50, E = 50, T = 0.5, D = 0.1, r = 0.03, \sigma = 0.7$, using the Monte-Carlo method and allowing the MC error to exceed 0.01 with probability $\alpha = 0.05$ (for the stock price generation, take $\mu = r - D$).

2. Study the strong convergence rate of the Euler's method experimentally. First write a function `S_euler_error(n,S0,m,T,mu,sigma)`, which returns the difference between the $S(T)$ obtained by the Euler's method and the exact value based on the formula

$$S(T) = S0 \cdot e^{(\mu - \frac{\sigma^2}{2})T + \sigma B(T)}$$

   on both instances i.e. when calculating the exact value of $S(T)$ one needs to set $B(T) = \sum_{i=1}^{m} X_i$, where X_i are the random variables used with Euler's method. Then use Monte Carlo method to estimate the expected value of the absolute difference. Use the same parameters than in the first task. To analyze the dependence on m calculate the expected value for $m = 5, 10, 20, 40, 80$ and the use the R command `nls(log(errors)~log(c)-q*log(m),start=list(c=1,q=1))` to estimate the rate of convergence. In that command `errosr` has to be the vector of expected values of errors and `m` has to be a vector of m values used in computations.

3. **Homework problem 2** (deadline September 27, 2020) Assume That Black-Scholes market model holds with $S0 = 68, T = 0.5, D = 0.03, r = 0.02, \sigma = 0.7$. Determine experimentally the weak rate of convergence of the Euler's method for the option with the payoff

$$p(s) = \max\{|s - 70| - 10, 0\}|.$$

   Since the option is equivalent to holding a call option with exercise price 80 and a put option with exercise price 60 and the volatility is constant, we can use BS formula for Call options to compute the exact value of the option of this problem. Use $m = 3, 6, 12, 24$ for determining the parameter q by generating $n = 10^6$ values of $S(T)$ for each computation. (Hint: Compute the exact option price and then, for each m, compute the difference between the exact price and the approximate price obtained by using MC1 with the generator `S_euler` in the case of m time steps)