

Computational Finance

Computer Lab 3

The aim of the Lab is to learn to simulate the paths of solutions of stochastic differential equations corresponding to common stock market models.

Sometimes (especially for applying Monte-Carlo methods) it is important to know how to simulate the stock price trajectories corresponding to a market model. A simple and quite universal (but often not the best) way to generate the trajectories of solutions of stochastic differential equations is Euler-Maruyama method, where differentials are replaced by differences over small time intervals (t_i, t_{i+1}) and all other values on the right hand side are taken at the time moment t_i . The same idea applies if we have one equation or many equations.

For Black-Scholes market model

$$dS(t) = S(t)(\mu(t) dt + \sigma(S(t), t) dB(t))$$

this leads to an approximation

$$S(t_{i+1}) - S(t_i) \approx S(t_i)(\mu(t_i) \Delta t + \sigma(S(t_i), t_i) (B(t_{i+1}) - B(t_i)))$$

where $0 = t_0 < t_1 < \dots < t_m = T$ is a partition of the interval $[0, T]$ into (usually equal) subintervals and $h_i = t_{i+1} - t_i$. If the time intervals are equal, we can use the single value $h = \frac{T}{m}$ instead of h_i . Using this approximation, the knowledge that $B(t_{i+1}) - B(t_i) \sim N(0, \sqrt{h_i})$ and a given value of $S_0 = S(0)$ we can compute approximate values $S_1, S_2, \dots, S_m$ of $S(t_1), S(t_2), \dots, S(t_m)$ by

$$S_{i+1} = S_i (1 + \mu(t_i) h_i + \sigma(S_i, t_i) \sqrt{h_i} X_i), \quad i = 0, \dots, m-1,$$

where X_i are independent random variables from the standard normal distribution.

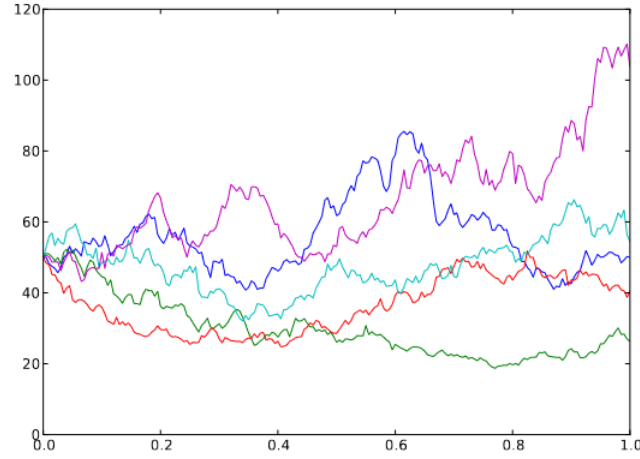
Exercise 1. Write a function **BSgraph(S0,n,m,mu,sigma,T)** that plots the graph of n trajectories of the stock price on the interval $[0, T]$, corresponding to the Black-Scholes market model with constant parameters μ and σ . For computing the values of the stock prices divide the interval $[0, T]$ into m equal subintervals (ie. use the time points $t_i = \frac{iT}{m}$, $i = 0, 1, \dots, m$) and use the Euler-Maruyama method. For this

1. Define a $(m+1) \times n$ matrix S to store the values of the stock prices, in each column the values of a different trajectory and in i -th row the values corresponding to the time moment t_i , $i = 0, 1, 2, \dots, m$.
2. All the trajectories start from the value S_0 that is given in the parameter $S0$ of the function, so define **S[0,:]=S0**.
3. Now, at each time moment generate a vector of different random numbers for each trajectory by the command **np.random.randn(n)** and compute the $(i+1)$ -th row of the matrix with a single line of the code by **S[i+1,:] =S[i,:]*(1+...**
4. Since this function does not have to return a value (it draws the graph instead), there does not have to be any return commands at the end of the function. So the function should end with **plt.plot(...)** and **plt.show()** commands (assuming the package **pyplot** of **matplotlib** has been imported with alias **plt**).

If the code is written correctly, then by entering the command

BSgraph(S0=50,n=5,m=200,mu=0.1,sigma=0.5,T=1)

a picture similar to the following should be generated:



Exercise 2. In the case of pricing European options by Monte-Carlo methods, we do not want to look at the trajectories of the stock prices but need only to generate values of $S(T)$. Define a function `ST(S0,n,m,mu,sigma,T)` that returns a vector of n randomly generated values of $S(T)$ according to BS market model with constant μ and non-constant volatility given by a function $\sigma(s, t)$. Compute the mean value and standard deviation of 100000 generated stock prices at time $t = T$ in the case $\mu = 0.05$, $m = 100$, $S_0 = 10$, $T = 1$ and

$$\sigma(s, t) = \frac{e^{-0.1 \cdot t}}{1 + 0.005s^2}.$$

For checking answers: mean and standard deviation should be approximately 10.5 and 6.1.

Exercise 3. Often we have several stochastic processes in a market model. Let us consider a model with stochastic interest rate:

$$\begin{aligned} dS(t) &= S(t)(r(t) dt + 0.5dB_1(t)), \\ dr(t) &= (0.05 - r(t)) dt + 0.02 dB_2(t), \end{aligned}$$

where B_1 and B_2 are independent Brownian motions. Use Euler-Maruyama method for defining a function that for given n and m outputs n generated values of $S(0.5)$ in the case $S(0) = 100$, $r(0) = 0.04$.