

Computational Finance, Fall 2018

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-1}, t_i) and all other values on the right hand side are taken at the time moment t_{i-1} . 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

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

where $0 = t_0 < t_1 < \dots < t_m = T$ is a partition of the interval $[0, T]$ into (usually equal) subintervals and $h_i = t_i - t_{i-1}$. 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 $B(t_i) - B(t_{i-1}) \sim N(0, \sqrt{h_i})$ 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

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

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

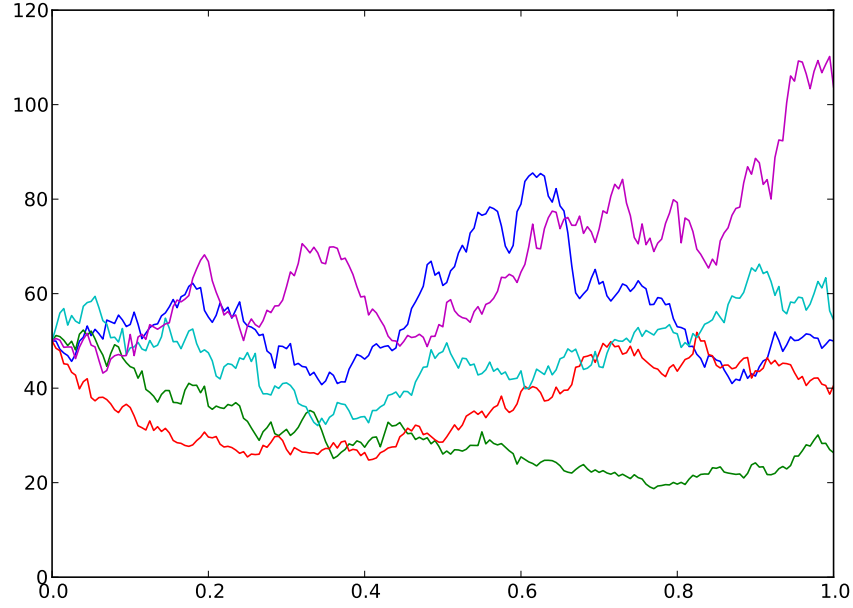
Exercise 1. Write a function `BSgraph(S0,n,m,mu,sigma,T)` that plots the graph of n trajectories of the stock price on the interval $[0, T]$, corresponding to the Black-Scholes market model with constant parameters μ and σ . For computing the values of the stock prices divide the interval $[0, T]$ into m equal subintervals (ie. use the time points $t_i = \frac{i \cdot T}{m}$, $i = 0, 1, \dots, m$) and use the Euler's method. This is again a good example when it is possible to use vectorized commands to make the code shorter (and to speed up computations). 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 , $i = 0, 1, 2, \dots, m$. All the trajectories start from the value S_0 that is given in the parameter $S0$ of the function, so we can write the value in the 0-th row of the matrix by using the command `S[0,:]=S0`. Now, according to the Euler-Maruyama method we compute the value of each trajectory at the next time moment by using the values of the stock prices at the previous time moment and a different random number for each trajectory. We can generate a vector of different random numbers for each trajectory by the command `np.random.randn(n)` and so for each $i = 1, 2, \dots, m$ we can compute the i -th of the matrix with a single line of the code by `S[i,:] = S[i-1,:]*(1+...`. We have to do this step by step (since the values of the previous row have to be computed before we can compute the next one, so using a for cycle over the values of i is necessary).

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 `pl.plot(...)` and `pl.show()` commands (assuming the package `pylab` has been imported with alias `pl`).

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 $ST(S_0, n, m, \mu, \sigma, T)$ 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 $\sigma(s, t) = \frac{e^{-0.1 \cdot t}}{1 + 0.005 s^2}$. (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:

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

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 $S(0) = 100$, $r(0) = 0.04$.

Homework problem 1. (Deadline September 27, 2018) It is well known that the Black-Scholes model is not perfect and that, if the model holds, then the volatility can not be constant for most stocks. A possible alternative is to allow the variance of the stock price changes to be stochastic, too. Let us consider the model

$$\begin{aligned} dS(t) &= S(t)(0.05 dt + \sqrt{V(t)} dB_1(t)), \\ dV(t) &= V(t) \cdot (0.25 - V(t)) dt + 0.1 V(t)^{1.5} dB_2(t), \end{aligned}$$

where B_1 and B_2 are independent Brownian motions. Define a function **Sgen** that for given m and n generates n stock prices $S(T)$ in the case $T = 0.5$, $S(0) = 50$, $V(0) = 0.2$ by using Euler-Maruyama method with m time steps. Using the function with $m = 100$, $n = 10000$, compute approximately the expected value of $\max(20 - |S(T) - 35|, 0)$.