Students: Emanuele Fogliati, Fabio Di Giovanni, Marco Laudone, Caterina Negri

# IT training project

ASSIGNMENT 2.3

## Contents

# 1. The financial and mathematical problem

The aim of this project is to simulate the dynamics of a portfolio composed by two stocks (which are correlated to one another) and a European call option on the second stock, which has maturity $T$.
The portfolio weights are:

- $\frac{n_1}{M} = \frac{4}{15}$ = portfolio share of the first stock, whose price at time 0 is equal to 100
- $\frac{n_2}{M} = \frac{3}{15}$ = portfolio share of the second stock, whose price at time 0 is equal to 85
- $\frac{n_3}{M} = \frac{8}{15}$ = portfolio share of the call option on the second stock

We assume that the two stocks follow a two-dimensional Geometric Brownian motion. The initial portfolio value $V_0$ is calculated in the following way:

$$V_0 = S_0^1 \cdot n_1 + S_0^2 \cdot n_2 + C \cdot n_3$$

Where $S_0^1$ is the price of the first stock at time 0, $S_0^2$ is the price of the second stock at time 0 and $C$ is the price of the call option under the risk-neutral measure at time 0.
After modelling the trajectory of each asset contained in the portfolio, its final value at time $T$, $V_1$, is calculated as:

$$V_1 = S_T^1 \cdot n_1 + S_T^2 \cdot n_2 + \max(S_T^2 - K, 0) \cdot n_3$$

Where $S_T^1$ is the price of the first stock at time $T$, $S_T^2$ is the price of the second stock at time $T$ and $\max(S_T^2 - K, 0)$ is the option payoff at time $T$ (which will be zero if $S_T^2$ is smaller than the strike price $K$ and equal to $S_T^2 - K$ otherwise). In our example, we have that the maturity of the option is $T = 1$.
We choose to split the interval $[0, T]$ in 10 sub-intervals, representing 10 days (we have one observation per day because we only consider the closing price to calculate the final value of the portfolio and the option payoff). Then, we may find the loss $L$, defined as:

$$L := -\Delta V = V_0 - V_1$$

Moreover, to evaluate the riskiness of the portfolio, we will compute the Value at Risk (VaR), which identifies the loss that the client would incur holding this portfolio for a period $[0, T]$ in the worst-case scenario (statistically, the VaR is represented by the $\varepsilon^{th}$ percentile of the cumulative distribution function of $L$).

In order to carry out this assignment, we will first simulate one trajectory for the price of each stock and then calculate its initial and terminal value (in order to find the loss) using the simulated prices of the stocks and the price of the option calculated thanks to the Black and Scholes exact pricing formula.
Then, to obtain the expected loss and the measure of risk (the VaR evaluated at 99%, 99.5% and 99.9% level) we will run a Monte Carlo simulation and generate multiple paths for each stock, which will be used first to calculate the simulated price of the call option and then to model the dynamics of our portfolio.

# 2. The solution

One of the most widely used models for stock prices is the Geometric Brownian Motion, which implies log-normal prices. As anticipated, in this project, since the two stocks are assumed to be correlated with one another, we suppose that the prices of the two assets follow a two-dimensional Geometric Brownian Motion.
Generally, a stochastic process $S_t$ is said to follow a Geometric Brownian Motion if it satisfies the following SDE:

$$\mathrm{d}S_t = S_t(r\mathrm{d}t + \sigma \mathrm{d}W_t)$$

where $W_t$ is a Brownian Motion. Thanks to Ito's formula, we can derive the solution to the SDE above:

$$S_t = S_0 \cdot \exp\left\{\left(r - \frac{\sigma^2}{2}\right) \cdot t + \sigma \cdot W_t\right\}$$

## Proof

We need to apply Ito's formula to the following process:

$$S_t = S_0 \cdot \exp\left\{\underbrace{\left(r - \frac{\sigma^2}{2}\right)}_{=:\mu} \cdot t + \sigma \cdot W_t\right\} = S_0 \cdot \exp\{\mu \cdot t + \sigma \cdot W_t\}$$

Setting $S_t = f(t,x) = S_0 \cdot \exp\left\{\left(r - \frac{\sigma^2}{2}\right) \cdot t + \sigma \cdot x\right\}$ we have that:

$$\frac{\partial f}{\partial t}(t,x) = \mu \cdot \underbrace{S_0 \cdot \exp\{\sigma x + \mu t\}}_{f(t,x)} \quad \frac{\partial f}{\partial x}(t,x) = \sigma \cdot \underbrace{S_0 \cdot \exp\{\sigma x + \mu t\}}_{f(t,x)} \quad \frac{\partial^2 f}{\partial x^2}(t,x) = \sigma^2 \cdot \underbrace{S_0 \cdot \exp\{\sigma x + \mu t\}}_{f(t,x)}$$

$$\frac{\partial f}{\partial t}(t,W_t) = \mu \cdot S_t \qquad \frac{\partial f}{\partial W_t}(t,W_t) = \sigma \cdot S_t \qquad \frac{\partial^2 f}{\partial W_t^2}(t,W_t) = \sigma^2 \cdot S_t$$

Therefore:

$$\mathrm{d}S_t = \frac{\partial f}{\partial t}(t,W_t)\mathrm{d}t + \frac{\partial f}{\partial W_t}(t,W_t)\mathrm{d}W_t + \frac{1}{2}\frac{\partial^2 f}{\partial W_t^2}(t,W_t)\underbrace{(\mathrm{d}W_t)^2}_{\mathrm{d}t}$$

$$\mathrm{d}S_t = (\mu \cdot S_t)\mathrm{d}t + (\sigma \cdot S_t)\mathrm{d}W_t + \frac{1}{2}(\sigma^2 \cdot S_t)\mathrm{d}t$$

$$\mathrm{d}S_t = \left[\left(\mu + \frac{1}{2}\sigma^2\right) \cdot S_t\mathrm{d}t + (\sigma \cdot S_t)\mathrm{d}W_t\right]$$

$$\mathrm{d}S_t = S_t\left[\left(r - \frac{\sigma^2}{2} + \frac{1}{2}\sigma^2\right)\mathrm{d}t + \sigma\mathrm{d}W_t\right]$$

$$\mathrm{d}S_t = S_t(r\mathrm{d}t + \sigma\mathrm{d}W_t)$$

Q.E.D.

In this project, we want to simulate the behaviour of two stocks which are correlated to one another: we assume that they are modelled by a two-dimensional Geometric Brownian motion, i.e. the solution to the following system of SDEs:

$$\mathrm{d}S_t^i = S_t^i\left(\mu^i\mathrm{d}t + \sigma^i\mathrm{d}W_t^i\right) \qquad i = 1,2$$

where each $W^i$ is a standard one-dimensional Brownian Motion and

$$\mathrm{Corr}[W_t^1, W_t^2] = \rho_{1,2} = 0.15, \qquad \forall\ t = 1, \dots, T$$

Its covariance matrix (which is positive definite and symmetric) is:

$$\mathbf{\Sigma} = \begin{bmatrix} \sigma_1^2 & \sigma_1\sigma_2\rho_{1,2} \\ \sigma_1\sigma_2\rho_{1,2} & \sigma_2^2 \end{bmatrix} = \begin{bmatrix} 0.2^2 & 0.2 \cdot 0.12 \cdot 0.15 \\ 0.2 \cdot 0.12 \cdot 0.15 & 0.12^2 \end{bmatrix} = \begin{bmatrix} 0.04 & 0.0036 \\ 0.0036 & 0.0144 \end{bmatrix}$$

The vector $\boldsymbol{\mu}$ can be defined as:

$$\boldsymbol{\mu} = \begin{bmatrix} r - \frac{\sigma_1^2}{2} \\ r - \frac{\sigma_2^2}{2} \end{bmatrix} = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix} = \begin{bmatrix} 0.02 \\ 0.0328 \end{bmatrix}$$

In this project, we want to model the behaviour of $\mathbf{S} = S_1, S_2$, which is a two dimensional $GBM(\boldsymbol{\mu}, \mathbf{\Sigma})$ ($S_1$ is a stochastic process representing the trajectory of the price of the first stock while $S_2$ is a stochastic process representing the trajectory of the price of the second stock). Of course, $S_1$ and $S_2$ satisfy the following system of SDEs:

$$\begin{cases} \mathrm{d}S_t^1 = S_t^1(\mu^1 \mathrm{d}t + \mathbf{A}\mathrm{d}W_t^2) \\ \mathrm{d}S_t^2 = S_t^2(\mu^1 \mathrm{d}t + \mathbf{A}\mathrm{d}W_t^1) \end{cases}$$

Where $\mathbf{A}$ satisfies $\mathbf{AA}' = \mathbf{\Sigma}$. In our case, thanks to Choleski's factorization, we have that:

$$\mathbf{\Sigma} = \begin{bmatrix} \sigma_1^2 & \sigma_1\sigma_2\rho_{1,2} \\ \sigma_1\sigma_2\rho_{1,2} & \sigma_2^2 \end{bmatrix} = \begin{bmatrix} \sqrt{\sigma_1^2} & 0 \\ \frac{\sigma_1\sigma_2\rho_{1,2}}{\sigma_1} & \sqrt{\sigma_2^2 - (\sigma_2\rho_{1,2})^2} \end{bmatrix} \begin{bmatrix} \sqrt{\sigma_1^2} & \frac{\sigma_1\sigma_2\rho_{1,2}}{\sigma_1} \\ 0 & \sqrt{\sigma_2^2 - (\sigma_2\rho_{1,2})^2} \end{bmatrix}$$
$$= \begin{bmatrix} \sigma_1 & 0 \\ \sigma_2\rho_{1,2} & \sigma_2\sqrt{(1-\rho_{1,2}^2)} \end{bmatrix} \begin{bmatrix} \sigma_1 & \sigma_2\rho_{1,2} \\ 0 & \sigma_2\sqrt{(1-\rho_{1,2}^2)} \end{bmatrix} \Rightarrow \mathbf{A} = \begin{bmatrix} \sigma_1 & 0 \\ \sigma_2\rho_{1,2} & \sigma_2\sqrt{(1-\rho_{1,2}^2)} \end{bmatrix}$$

On the other hand, the price and the payoff of the option depends on the behaviour of the second stock and the strike price $K$. The European call option has maturity $T$ and terminal payoff equal to:

$$D = \max(S_T^2 - K, 0) = (S_T^2 - K)^+$$

Under the risk-neutral measure, the price of the call option is the discounted value of the expected payoff:

$$C = e^{-rT} \cdot \mathbb{E}[(S_T^2 - K)^+]$$

We can also obtain a closed form solution that gives us the exact formula for the price of the call option:

$$C^E = S_0^2 \cdot \Phi(d_1) + e^{-rT} \cdot K \cdot \Phi(d_2)$$

Where:

$$d_1 = \frac{\log\left(\frac{S_0^2}{K}\right) + \left(r + \frac{1}{2}\sigma_2^2\right)T}{\sigma_2\sqrt{T}}$$
$$d_2 = \frac{\log\left(\frac{S_0^2}{K}\right) + \left(r - \frac{1}{2}\sigma_2^2\right)T}{\sigma_2\sqrt{T}}$$

And $\Phi(.)$ is the cumulative distribution function of a Standard Normal random variable.

# 3. The numerical algorithms

After generating random variables, we can focus on the construction of paths.

Generating sample paths amounts to discretizing the process, i.e., approximating it by considering its realizations at a finite set of time points $[t_0, t_1, \dots, t_N]$ and then interpolate to produce a continuous-time trajectory. It is usually wise to split the sampling interval into equally spaced sub-intervals.

In our case, samples from continuous diffusions are generated using the closed-form solution of the SDE (as we have previously said), i.e., the exact dynamics followed by the process.

The method of "exact solution" can be applied if the SDE can be solved explicitly, i.e., if there exists a functional of time $t$ and the driving noise $W$ up to t such that:

$$X_t = G(t, \{W_i\}_{i=0,\dots,t})$$

It consists in discretizing the underlying noise over a finite set of sampling times and then applying the functional to obtain the value of the process $X$ at those set of time points.

Algorithm:

1. Set $X_0 = x_0$ and $\Delta t = T/N$
2. For $i = 1, \dots, N$, sample the Brownian motion $W(t_i)$ and set $X_i = G(t_i, \{W_{t_1}, \dots, W_{t_i}\})$
3. $(X_i)_{i=1,\dots,N}$ is a sample of the process $X$ on $[0, T]$

A multidimensional Geometric Brownian motion can be specified as the system of SDEs:

$$\mathrm{d}S_t^i = S_t^i\left(\mu^i \mathrm{d}t + \sigma^i \mathrm{d}W_t^i\right)$$

where each $W^i$ is a standard one-dimensional Brownian motion and $W_t^i$ and $W_t^j$ have instantaneous correlation $\rho_{ij}, i = 1, \dots, d$. Taking $\sum_{ij} = \sigma_i\sigma_j\rho_{ij}$, we say that $S = S^1, S^2, \dots, S^d$ is a $d$-dimensional GBM($\mu, \sum$).

Then:

$$\frac{dS_t^i}{S_t^i} = \mu^i \mathrm{d}t + A\mathrm{d}W_t^j$$

With A being such that $\mathbf{AA'} = \mathbf{\Sigma}$

If we discretize it, we get:

$$S_{t_{k+1}} = S_{t_k} e^{\left(\mu - \frac{1}{2}\sigma^2\right)\Delta t + AZ_{k+1}}$$

where $Z_k \sim N(0, \mathbf{I})$, $\forall\, Z_1, \dots, Z_n$ independent and $\sigma^2 = \mathrm{diag}(\mathbf{\Sigma})$. The multidimensional Geometric Brownian motion is very useful to model the evolution of stock prices, considering their correlations, and this why we implement this algorithm to simulate the behaviour of this particular portfolio.
In particular, we will first generate just one trajectory for each stock, and then store it into a $2 \times N$ matrix S, whose columns represents the observations at each time interval $t_k$ and the rows

represent the two stocks.

In the last point of this assignment, we will instead generate a number `NPATHS` of trajectories for each stock the three-dimensional matrix Simulated (in particular, a $2 \times N \times NPATHS$ matrix). This will be done following this algorithm:

1. Set $\begin{bmatrix} S_0^1 \\ S_0^2 \end{bmatrix} = \begin{bmatrix} s_0^1 \\ s_0^2 \end{bmatrix} = \begin{bmatrix} 100 \\ 85 \end{bmatrix}$ (i.e, set the initial prices of stock 1 and stock 2 equal to 100 and 85 respectively) and $\Delta t = T/N$
2. Generate a $2 \times N \times NPATHS$ matrix of values sampled from a standard Normal with the command `randn`
3. For $i = 1, \dots, N$, sample the Brownian motion $W(t_i)$ and set $S_i = G(t_i, \{W_{t_1}, \dots, W_{t_i}\})$ for each trajectory (as explained below)
4. We use two nested for loops to fill in the three-dimensional matrix `Simulated`. The first for loop has `NPATHS` iterations (at each iteration, one sample path is generated for both assets). The second nested for loop has $N + 1$ iterations, and at each iteration we generate two values of the correlated GBM (one for each asset) which represent the value assumed at each discrete time interval, using:
$$\begin{bmatrix} S_{t_{k+1}}^1 \\ S_{t_{k+1}}^2 \end{bmatrix} = \begin{bmatrix} S_{t_k}^1 \\ S_{t_k}^2 \end{bmatrix} \exp\left\{\left(\boldsymbol{\mu} - \frac{1}{2}\boldsymbol{\sigma}^2\right)\Delta t + \mathbf{A}Z_{k+1}\right\}$$
Where $Z_{k+1}$ is a column vector containing two values sampled from a Standard Normal distribution, $\boldsymbol{\sigma}^2 = \operatorname{diag}(\boldsymbol{\Sigma})$ and $\boldsymbol{\mu} = \begin{bmatrix} r - \frac{\sigma_1^2}{2} \\ r - \frac{\sigma_2^2}{2} \end{bmatrix}$
5. For each trajectory generated, we have that $\begin{bmatrix} S_i^1 \\ S_i^2 \end{bmatrix}_{i=1,\dots,N}$ is a sample of the correlated GBM $\mathbf{S}$ on the time interval $[0, T]$, discretized in $N$ intervals $\mathrm{d}t$ (in our case, we have 10 intervals for each trajectory).

To model the behaviour of our portfolio, we also have to consider the pricing of a European call option in a Black Scholes setting.

The option payoff is $(S_T^2 - K)^+$, where $S_T^2$ is the price of stock 2 at time $T$ (maturity of the option) and $K$ is the strike price. As we have specified before, the stochastic process $S$ follows the SDE:
$$\frac{dS_T}{S_T} = r\mathrm{d}t + \sigma \mathrm{d}W_t$$
under the so-called risk-neutral measure. The price of the option is the expected discounted value (under a particular measure, the risk-neutral one) of its terminal payoff:
$$C = e^{-rT} \cdot \mathbb{E}[(S_T - K)^+]$$

The fact that derivatives prices (which are solution to PDEs) can be written as expectations (via the Feynman-Kac formula) allow us to state a link between derivatives pricing and Monte Carlo simulation.

Indeed, the price of the call can be estimated by the following algorithm:

1. For $i = 1, \dots, n$ generate asset prices at T, i.e., generate $Z_i \sim N(0,1)$

2. Compute $S_T^i = S_0 \exp\left\{\left(r - \frac{1}{2}\sigma^2\right)T + \sigma\sqrt{T}Z_i\right\}$
3. Set $C_i = e^{-rT}(S_T - K)^+$
4. Compute $\overline{C_n} = \frac{C_1 + \ldots + C_n}{n}$

We know that by LLN and CLT $\overline{C_n}$ is an unbiased estimator of the price $C$ of the Call and that as $n \to \infty$, we have that $\overline{C_n} \to C$ with probability 1.

In our project, we implement the following algorithm:

1. Since we have already sampled from a Standard Normal `NPATHS` times and stored it in the matrix `Z2`, we don't need to generate the values again. Therefore, we use the vector of values in position $(:, N, :)$ (i.e., the last value generated for each trajectory for asset 2 and asset 1) in the matrix `Z2` to implement the algorithm described above. We obtain:
$$\mathbf{Z}_i = \begin{bmatrix} Z2(1, N+1, i) \\ Z2(2, N+1, i) \end{bmatrix}, \qquad i = 1, \ldots, NPATHS$$
2. We run a `for loop` with `NPATHS` iterations to run the Monte Carlo simulation necessary to obtain the Monte Carlo price of the call option. First, we compute the terminal values both for stock 1 and stock 2, using
$$\boldsymbol{S}_T^i = \begin{bmatrix} S_T^1 \\ S_T^2 \end{bmatrix} = \begin{bmatrix} S_0^1 \\ S_0^2 \end{bmatrix} \exp\{\boldsymbol{\mu} \cdot T + \mathbf{A}\mathbf{Z}_i \cdot \sqrt{T}\}, \qquad i = 1, \ldots, NPATHS$$
3. We only use $S_T^2$ to compute the discounted payoff at each iteration $C_i$ $(i = 1, \ldots, NPATHS)$:
$$C_i = e^{-rT}(S_T^2 - K)^+$$
4. Doing so, we obtain $NPATHS$ values for the discounted payoff. The Monte Carlo price of the option is simply:
$$C^{MC} = \frac{1}{NPATHS} \sum_{i=1}^{NPATHS} C_i$$

Finally, we use the vectors obtained in the previous points to compute the portfolio total value.
In point (4), we chose to compute the value of the portfolio that we would have if we used the two trajectories (one for each stock) generated in point (3). Since we only have one value to compute the final payoff of stock 2, we thought that it would be more appropriate to use the exact Black and Scholes price $C^E$. Therefore, in point (4), the loss $L$ is:

$$V_0 = n_1 \cdot S_0^1 + n_2 \cdot S_0^2 + n_3 \cdot C^E$$
$$V_1 = n_1 \cdot S_T^1 + n_2 \cdot S_T^2 + n_3 \cdot \max(S_T^2 - K, 0)$$
$$L = V_0 - V_1$$

In point (5), we have run a Monte Carlo simulation with the method described above both to compute `NPATHS` trajectories for each asset (so we obtain `NPATHS` terminal values for each stock) and also to compute the Monte Carlo price of the call option. Therefore, in point (5) we may construct two vectors of length `NPATHS` containing all the initial and final simulated values of our portfolio.

$$\mathbf{V}_0 = \begin{bmatrix} V_0^1 \\ V_0^2 \\ \vdots \\ V_0^{NPATHS} \end{bmatrix}$$

Of course, each component of the vector $\mathbf{V}_0$ (`V_initial_simulated`) is calculated as such:

$$V_0^j = n_1 \cdot S_0^1 + n_2 \cdot S_0^2 + n_3 \cdot C^{MC} \text{ with } \quad j = 1, \dots, NPATHS.$$

The vector containing all the final values of the portfolio (`V_final_simulated`) is:

$$\mathbf{V}_1 = \begin{bmatrix} V_1^1 \\ V_1^2 \\ \vdots \\ V_1^{NPATHS} \end{bmatrix}$$

Where:

$$V_1^j = n_1 \cdot S_T^1 + n_2 \cdot S_T^2 + n_3 \cdot \max(S_T^2 - K, 0), \quad j = 1, \dots, NPATHS$$

We then create the vector containing all the simulated losses:

$$\mathbf{L} = \mathbf{V}_0 - \mathbf{V}_1$$

$$\mathbf{L} = \begin{bmatrix} L_1 \\ L_2 \\ \vdots \\ L_{NPATHS} \end{bmatrix} = \begin{bmatrix} V_0^1 - V_1^1 \\ V_0^2 - V_1^2 \\ \vdots \\ V_0^{NPATHS} - V_1^{NPATHS} \end{bmatrix}$$

And then we use the vector $\mathbf{L}$ (`Loss_simulated`) to find the Value at risk (using the built-in function `quantile`) and the expected loss, which is just the sample mean:

$$\text{Expected Loss} = \frac{1}{NPATHS} \sum_{i=1}^{NPATHS} L_i$$

# 4. Code scripts

## Point 3

Simulating one trajectory for each asset

```
% Data
S1_0 = 100;
S2_0 = 85;
S_0 = [S1_0; S2_0];

r = 0.04;
sigma1 = 0.2;
sigma2 = 0.12;

rho12 = 0.15;

Sigma=[sigma1^2 sigma1*sigma2*rho12;
sigma1*sigma2*rho12 sigma2^2];
s = diag(Sigma);

A=[sigma1 0;
rho12*sigma2 sqrt(1-rho12^2)*sigma2];

mu1 = r-sigma1^2/2;
mu2 = r-sigma2^2/2;
mu = r-s./2;
T=1;
%Ten days (one observation per day, so N=10, T=1 so that dt = 0.1)
N=10;
dt=T/N;

% Simulating one path for the correlated GBM
S = zeros(2,N+1);
% Since we have two correlated GBM (representing two stocks) each row represents one GBM
and each column represents one observation
Set the first observations equal to S_0:
S(:,1) = S_0;

%Now we generate the matrix of standard Normals we'll use to compute dWt
Z = randn(2,N);

% Generating one path of the two correlated Brownian motions and storing it into a matrix
for i=1:N
    S(:,i+1) = S(:,i).*exp((mu(:,1)-s(:,1)/2)*dt+A*Z(:,i));
end
t=0:dt:T;
plot(t,S,'*-')
title('Stock 1 and 2 - Closing prices');
```

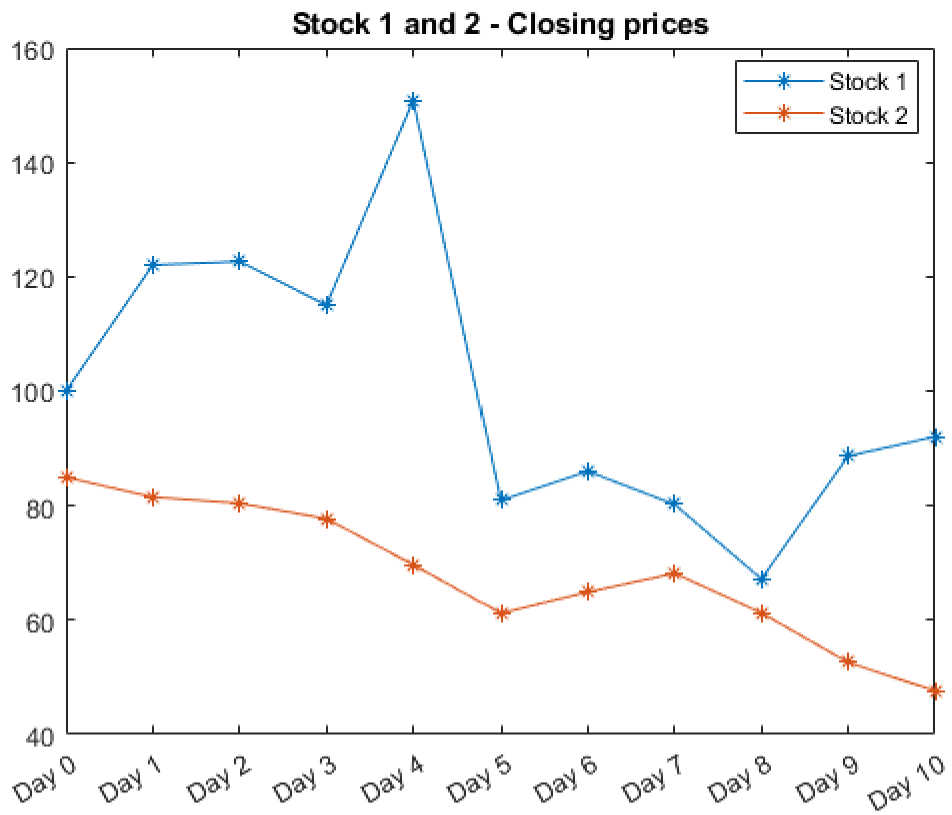
```
lgd = legend('Stock 1', 'Stock 2','boxoff');
set(lgd,'Location','Best');
xticks(t);
xticklabels({'Day 0','Day 1', 'Day 2','Day 3','Day 4','Day 5', 'Day 6','Day 7','Day
8','Day 9','Day 10'});

%The strike price of the option is:
K = 95;
% The exact price of the stock, according to the BS formula is:
d1=(log(S2_0/K)+(r+0.5*sigma2^2) * T)/(sigma2 *sqrt(T));
d2=(log(S2_0/K)+(r-0.5*sigma2^2) * T)/(sigma2 *sqrt(T));
Phi_d1=normcdf(d1);
Phi_d2=normcdf(d2);

BS_exact_price=S2_0*Phi_d1 -exp(-r*T)*K*Phi_d2;
BS_exact_price
```

```
BS_exact_price =

    1.7999
```



## Point 4

Computing the value of the portfolio (with the trajectories generated above)

```
n1 = 4;
n2 = 3;
n3 = 8;

V_initial_exact =  S(1,1)*n1+S(2,1)*n2+BS_exact_price*n3; %V0
V_final_exact = S(1,N+1)*n1+S(2,N+1)*n2+max(0,S(2,N+1)-K)*n3; %V1

Loss_one_trajectory = V_initial_exact-V_final_exact;
```

## Point 5

Monte Carlo simulation

```
NPATHS = 1000;
Simulated = zeros(2,N+1,NPATHS);
S_01 = zeros(2,NPATHS);
%Setting the initial values of each path equal to 100 for the first stock and 85 for the
second one:
S_01(1,:)= S1_0;
S_01(2,:) = S2_0;
Simulated(:,1,:) = S_01;
%Simulating a three-dimensional matrix containing values sampled from a standard Normal
(this is equivalent to simulating NPATHS matrices Z~N(0,I):
Z2 = randn(2,N,NPATHS);
%Filling in the matrix Simulated:
for j=1:NPATHS
    for i=1:N
        Simulated(:,i+1,j) = Simulated(:,i,j).*exp((mu(:,1)-s(:,1)/2)*dt+A*Z2(:,i,j));
    end
end

%This is the vector where we will store the simulated payoffs of the call option
C = zeros(1,NPATHS);

% Since we need only the last value of the path (ST), we can pick the last
% column of the matrix Z2 for each simulated path
for i=1:NPATHS
    temp = S_0.*exp(mu*T+A*Z2(:,N,i)*sqrt(T));
    C(1,i)=exp(-r*T).*max(0,temp(2,1)-K);
end

% For each path, C contains the payoff of the call option (on the second
% stock)
% The estimated MC price of the option is:
BS_MC_price = mean(C);
```

```
%The estimation error is:
MC_est_error = abs(BS_MC_price-BS_exact_price);

% The variance of the MC price is:
BS_MC_price_std = std(C);

% The standard error is:
BS_MC_price_std_err = BS_MC_price_std/sqrt(length(C));

%Creating the vectors of initial and final portfolio values
V_initial_simulated(:,1) = Simulated(1,1,:)*n1+Simulated(2,1,:)*n2 + BS_MC_price*n3;
V_final_simulated(:,1) = Simulated(1,N+1,:)*n1+Simulated(2,N+1,:)*n2 +
max(Simulated(2,N+1,:)-K,0)*n3;

%Creating the vector containing all the simulated losses
Loss_simulated = V_initial_simulated - V_final_simulated;

%Computing the expected loss:
expected_loss = mean(Loss_simulated);

%Calculate the VAR (using built-in functions)
VaR99=quantile(Loss_simulated,0.99);
VaR995=quantile(Loss_simulated,0.995);
VaR999=quantile(Loss_simulated,0.999);
```

# 6. Discussion of the results

A **European call option** gives its holder the right to purchase an asset for a specified price, called the **strike price,** on some specified expiration date. The holder of the call is not required to exercise the option. She will choose to exercise it only if the market value of the underlying asset, i.e., the second stock in our case, exceeds the strike price. In that case, the option holder may "call away" the asset for the strike price. Otherwise, the option may be left unexercised. If it is not exercised before the expiration date of the contract, a call option simply expires and becomes valueless. Therefore, if the stock price is greater than the strike price on the expiration date, the value of the call option equals the difference between the stock price, $S_T$, and the strike price, $K$; but if the stock price is less than the strike price at expiration, the call will be worthless.

$$Value\ at\ expiration = Stock\ Price - Strike\ Price = S_T^2 - K$$

The *net profit* on the call is the value of the option minus the price originally paid to purchase it. The purchase price of the option is called the **premium.** It represents the compensation the purchaser of the call must pay for the right to exercise the option only when exercising is desirable. Nevertheless, in order for the call buyer to clear a net profit, it is not sufficient that the stock price exceeds the strike price at the expiration date, it is also necessary that the stock price exceeds the strike price by more than the original investment made to buy the option, i.e., the exact price computed by Black Scholes.

In our case, at the strike price $K = 95$ we computed the exact purchase price of the European call option on the second stock following the Black Scholes formula. Under this result, the call buyer will clear a positive net profit, i.e. she will exercise the call option if and only if the following holds:

$$Final\,Value\ > Original\ investment$$

In our case, the original investment is equal to the price we paid to purchase the call option, i.e. $C$, and by using the Black Scholes formula the exact price of the European call option is:

$$C^E = 1.7999$$

By generating one path of the two correlated Brownian motions and store it in the matrix S we can plot and see how the second stock's price changes along the 10-day horizon and therefore if the value of the option at the expiration date is worthy or not.

Under Monte Carlo simulation the choice of exercising the call option or not depends indeed on the three-dimension matrix of standard normal random variables we generated, which is related to the chosen value of NPATHS:

- For NPATHS = 1000 the purchase price of the call option is: $C^{MC}$ = 1.880 (the estimation error is therefore 0.0802), and the standard error of the Monte Carlo estimate is $std.err(C^{MC})$ = 0.1383. In this case, the call buyer will clear a strictly positive net profit if and only if its final value > 1.880
- For NPATHS = 10000 the purchase price of the call option is: $C^{MC}$ = 1.775 (the estimation error is therefore 0.0250), and the standard error of the Monte Carlo estimate is $std.err(C^{MC})$ = 0.0422. In this case the call buyer will clear a strictly positive net profit if and only if its final value > 1.775
- For NPATHS = 100000 the purchase price of the call option is: $C^{MC}$ = 1.7938. (the estimation error is therefore 0.0061), and the standard error of the Monte Carlo estimate is

$std.err(C^{MC}) = 0.0136$. In this case the call buyer will clear a strictly positive net profit if and only if its final value > 1.7938

The final value of the simulated portfolio depends heavily on the fact that the call option has been exercised or not, i.e., if the call buyer exercises the option, she will have 8 more units of the second stock in her portfolio multiplied by the corresponding option payoff (`max(Simulated(2,N+1,:)-K,0)`) increasing the total final value of the portfolio. Indeed, if $Simulated(2, N+1, :) > K$ the value of the portfolio will be higher.

The simulated loss $L \coloneqq -\Delta V = V_0 - V_1$ will be greater the more the simulated values of the first and second stock prices at N+1, i.e. $S_T^1, S_T^2$ will exceed the initial stock prices $S_0^1 = 100, S_0^2 = 85$.

Under Monte Carlo simulation:

- For NPATHS = 1000 the expected loss is $L \coloneqq -201.5412$
- For NPATHS = 10000 the expected loss is $L \coloneqq -213.8086$
- For NPATHS = 100000 the expected loss is $L \coloneqq -213.8239$

This means that the final value of the portfolio, at the end of the 10-day horizon, is larger than its initial value:

$$V_1 > V_0$$

We conclude that, on average, the portfolio will make a profit.

As $NPATHS \to \infty$ the simulated price for the call option goes towards the exact price computed using the Black Scholes formula and the expected loss goes towards the loss computed under one trajectory of the Brownian motion.

$$\text{BS\_MC\_Price} \xrightarrow[NPATHS \to \infty]{} \text{S\_exact\_price}$$

$$\text{expected\_loss} \xrightarrow[\text{NPATHS} \to \infty]{} \text{Loss}$$

To assess the riskiness of a portfolio, we focus on the simulated distribution of these expected losses. The **value at risk** is the loss corresponding to a very low percentile of the entire return distribution of the expected loss. In our case we considered the 1st , 0.5th and 0.1st percentiles. VaR is written into regulation of banks and closely watched by risk managers. It is another name for *quantile* of a distribution. The quantile, $q$, of a distribution is the value below which lies $q$% of the possible values. Thus, if we estimate the 1% VaR, meaning that 99% of returns will exceed the VaR, and 1% will be worse. Therefore, the 1% VaR may be viewed as the best rate of return out of the 1% *worst-case* future scenarios.

For NPATHS = 1000 we have the following results:

$$1\%\ VaR = 412.08$$

$$0.5\%\ VaR = 436.14$$

$$0.1\%\ VaR = 466.45$$

The interpretation of these results is the following: 99% of the times, the loss we expect over the period $[0, T]$ is less or equal to 412.08, 99.5% of the times, the loss we expect over the period $[0, T]$ is smaller or equal than 436.14, and with a 99,9 % probability the expected loss at time $T = 1$ is equal to 466.45 (or smaller).