Problem Set 4

- This homework consists of two questions, each carrying equal marks. Your code will be graded by a Python script which compares your result against the baseline result. It is therefore important that your Python stack is identical to the baseline so that you avoid losing marks even though your code is correct.

1. Question 1

The following Brennan and Schwartz two-factor model describes the evolution of the short-rate $r(t)$ and the long rate $\theta(t)$

$$dr(t) = (a_1 + a_2(\theta(t) - r(t)))dt + \sigma_1 r(t) dW_t^{[1]}$$

and

$$d\theta(t) = \theta(t)(b_1 + b_2 r(t) + b_3\theta(t))dt + \sigma_2\theta(t)dW_t^{[2]}$$

and the two Wiener processes are correlated with correlation $\rho_{1,2}$. Put this function in `MonteCarloBrennanSchwartz.py`. Write a function

```
def BrennanSchwartzPaths(r0, theta0, a1,a2, b1, b2, b3, sigma1,sigma2, rho12,
        fixingTimes, samples):
```

that uses the usual Euler "step forward" method of forming Brennan Schwartz model short and long rate paths, where

- '`r0`' is the time `t` value of the short rate
- '`theta0`' is the time `t` value of the long rate
- '`sigma1`' is the constant volatility of the short rate
- '`sigma2`' is the constant volatility of the long rate
- '`a1`' is the constant drift of the short rate
- '`a2`' is the mean reversion rate of the short rate
- '`b1,b2,b3`' are constant drift coefficients of the long rate
- '`rho12`' is the correlation between the wiener processes for the long and short rates
- '`fixingTimes`' is the size N list of times relevant to the path
- '`samples`' is a two-dimensional $M \times N$ array of random uniform ($U[0, 1)$) samples.

This function should return a dictionary with the following two keys:

- '`paths_r`' an $M \times N$ `array` consisting of M short rate paths.
- '`paths_theta`' an $M \times N$ `array` consisting of M long rate paths.

2. Question 2

Evaluate the 99% quantile of the daily portfolio loss distribution for an equally weighted portfolio with n correlated risky assets using MC simulation. The correlation of asset returns is given by $\rho_{ij} = 0.2, i \neq j$. Each asset daily return is $N(\mu, \sigma^2)$ where $\mu = -0.1$ and $\sigma = 0.1$. You should implement the following function

```
def portfolioLossSimulation(mu, Sigma, w, alpha, samples):
```

where

- `'mu'` is the $n-$vector of daily asset returns means
- `'Sigma'` is the covariance matrix of daily asset returns
- `'samples'` is a two-dimensional $M \times n$ array of random uniform $(U[0, 1))$ samples.

The function should return the quantile estimate. Compare your MC estimate with the exact quantile and plot the convergence of the MC simulation to the exact quantile as the number of simulations M is increased.