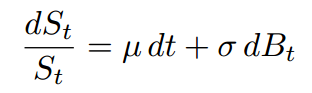
**Report**

Task 1(a)(i) : Simulation Geometric Brownian Motion (GBM)

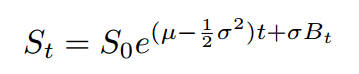
A simple model for stock prices can be obtained by assuming the return over infinitesimal dt is normally distributed with mean and variance both proportional to dt.



By assuming that the return has a stochastic differential equation (SDE) as following:



The solution of the SDE can be found by using Ito’s lemma:



Where,

represents the initial stock price,

, are parameters,

t is the end time,

is the Brownian motion at time t.

This solution is used to simulate 1000 runs of GBM for 0 < t < 3. The time [0,3] is partitioned into 1000 subintervals, each with length 0.003. 5 realizations of the GBM are plotted and is shown in Figure 1:

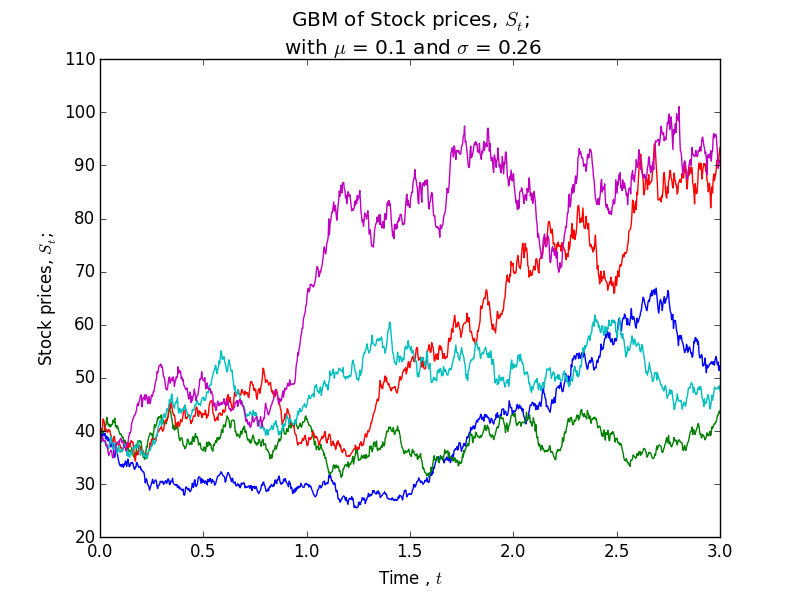


Figure 1

The theoretical expectation value of can be found by using the property of Brownian motion that a Brownian motion follows a normal distribution:

52.6444934955

The theoretical variance can also be found by using the formula of variance and also the property stated above:

Task 1(a)(ii) Finding expectation and probabilities

In the coding of *gbm.py*, each column of array S represents the stock prices for 1000 runs on each point of time and the last column of array S represents the stock prices for 1000 runs on time 3. The method of finding the expectation value of stock price at time 3, or , based on the simulation is similar to finding the mean; the built-in function *np.mean* is apply on the last column of array S. Variance can also be found with the built-in function *np.var.*

To find the probability of the more than 39, or P(> 39), a method called mask is used. In the code, the line *mask = S3 > 39* modifies every element in S3, where S3 is an array consist of 1000 different stock prices at time 3, and mask is a zero array of the same size as S3. If an element in S3 is more than 39, the corresponding position in mask will be 1, otherwise, the value of the position is remained. Every element in mask is then summed up and divided by the number of runs (n = 1000 in this case) to get the probability.

Lastly, the expected value of given > 39 is also calculated with the mask method and the calculation is as below:

=

Task 1(b)(i) : Simulating mean reversal process

Mean reversal process is a random process such that it tends to go towards the mean over time. To simulate this process, Euler-Maruyama scheme is using to approximate the solution numerically with Python codes and the SDE of the process.

Where:

are parameters,

is each time step ,

is the increment of Brownian motion in each t, created with normal random number generator in Python.

Even though the exact solution can be found, it is not used to simulate the process as the solution contains Ito integral that involves the past values of, which is tough to be implemented into the coding.

The simulation of the process, for 0 < t < 1 is done using the coding in file *mr.py* and 5 realizations of the mean reversal process is plotted as follows:

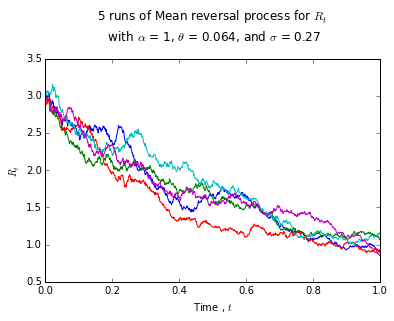


Figure 2

From Figure 2, it can be observed that the processes decrease over time and tend to a certain range of values.

Task 1(b)(ii) : Expectation and probability of mean reversal process

The expectation value of, at time 1 (or known as) is obtained by taking the mean of all the 1000 values in the last column of the array *R* in the code, which represents the value of in 1000 different runs, with the built-in function *np.mean*. While the probability of greater than 2 can be calculated by the mask method shown in the simulation of Geometric Brownian motion above.

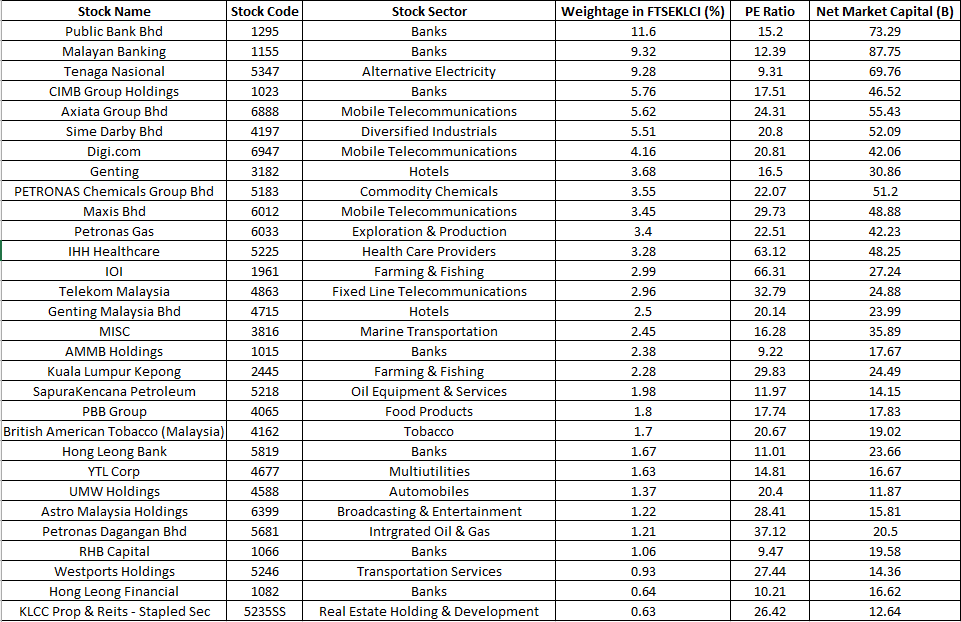
The following are the values obtained from the data that simulates Figure 2:

= 1.15508068326 = 0.011

Task 2(i) : Downloading stock data

The FTSE Bursa Malaysia KLCI is composed of the top 30 companies by market capitalization on the Bursa Malaysia Main market. It is reviewed semi-annually in June and December to ensure the Indices remain representative of the underlying Malaysia market.

The following table contains detailed information about the 30 component stocks in FTSEKLCI:



Task 2(ii): Finding moving average and correlation

A moving average is a type of technical analysis that calculates the average value of a stock’s price over time. To calculate the 5-day moving average of a stock in Python (1295.KL or Public Bank Berhad in this case), the built-in function *pd.rolling\_mean* can be used. In the beginning, stock prices of 1295.KL from 1st February 2012 to 20th July 2015 are downloaded and stored into a variable *closing\_data*. Then, the moving average is calculated and is plotted along with the stock price in Figure 3 below:

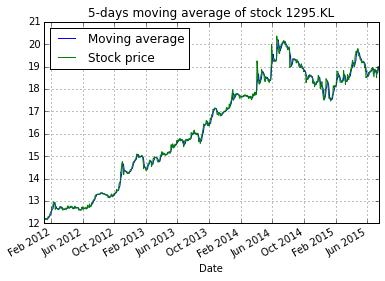


Figure 3

Python has a built-in function to calculate correlation between the stock price of Public Bank Berhad (PBB) and also the price of FTSEKLCI. This function is included in the module *pandas*. The stock prices of PBB and FTSEKLCI is first collected and combined into a DataFrame variable called *closing*. There are some days where the stock PBB exist but not the case for FTSEKLCI. To ease the calculation of correlation, the price of FTSEKLCI for those days are assumed to be zero. Next, the correlation is then calculated with the function *closing.corr()* . The correlation between PBB and FTSEKLCI obtained from the data collected is 0.911704, which means the stock prices of PBB and FTSEKLCI are highly positive correlated to each other.