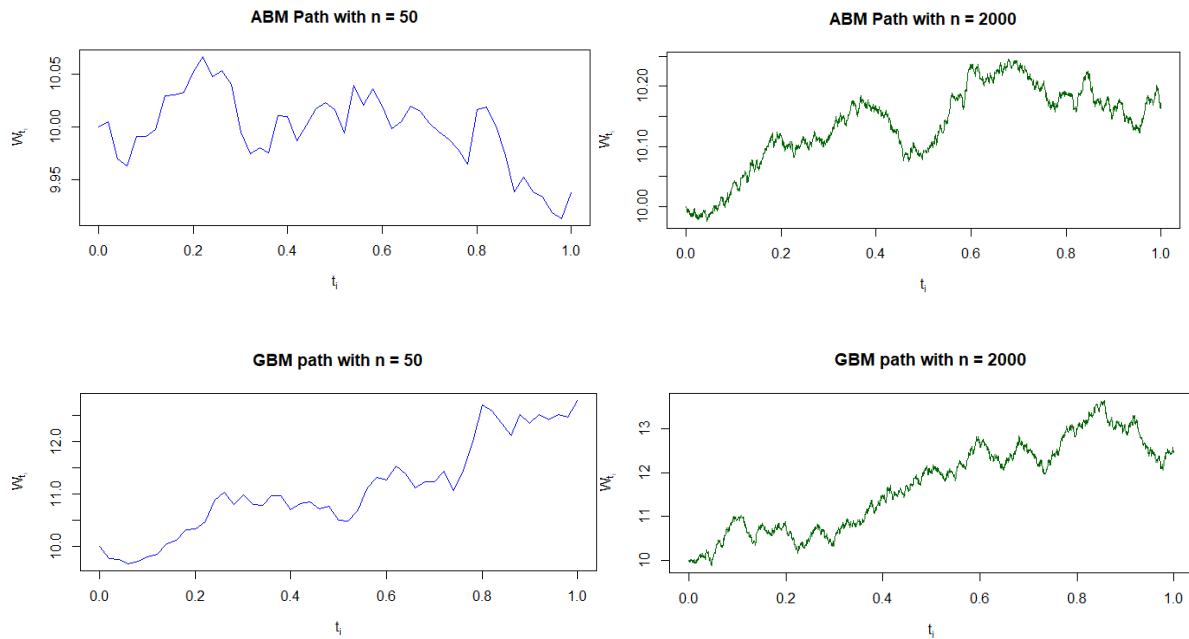


PROJECT 2 REPORT



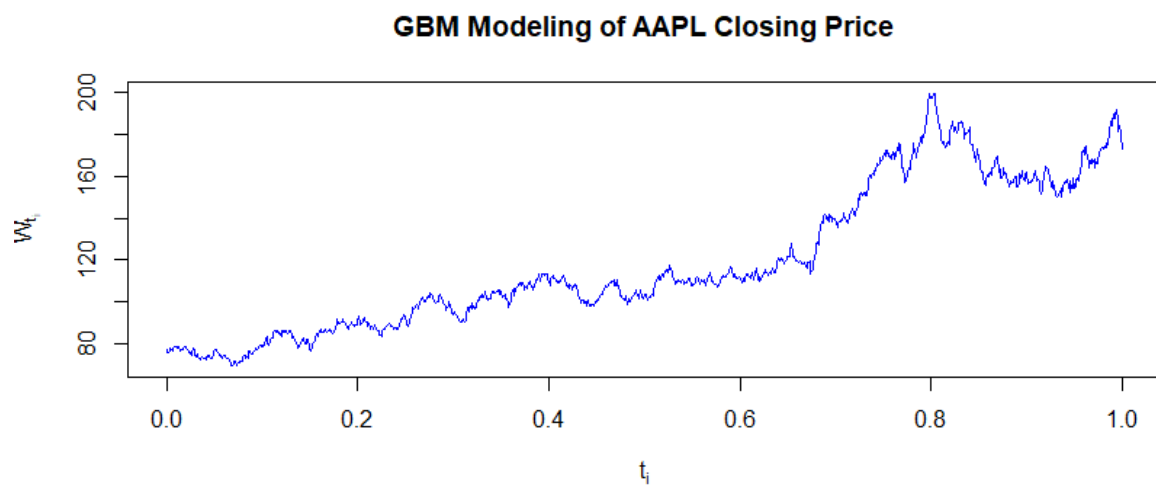
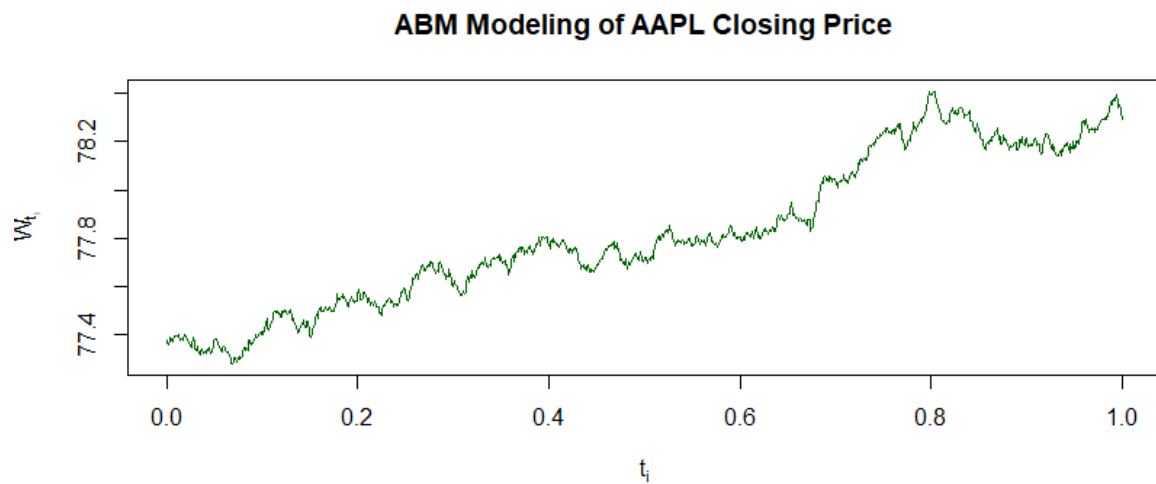
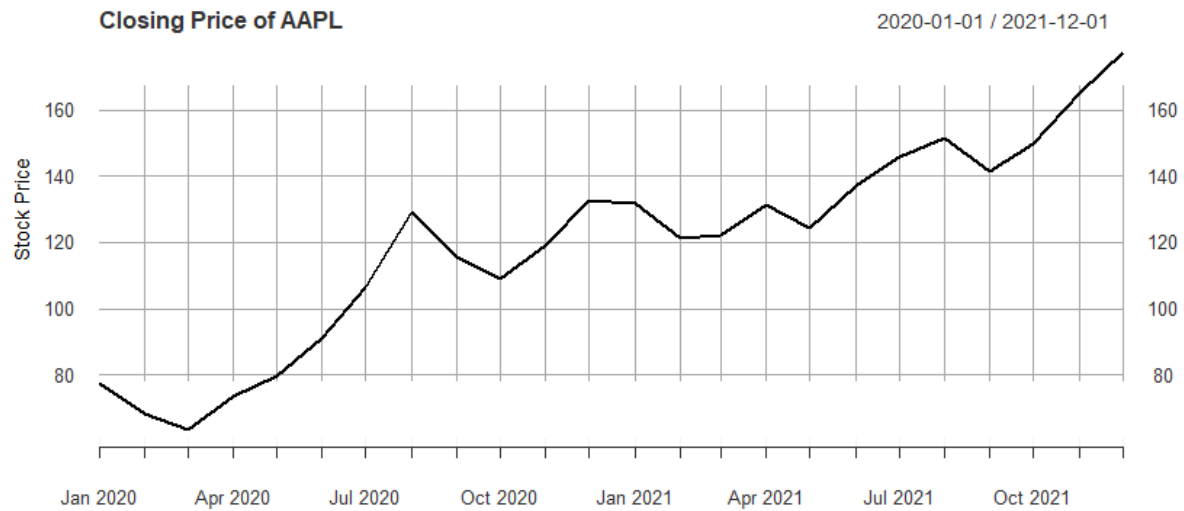
Comment:

The higher the value of n , the smaller the value of time change $dt = \frac{T}{n}$. It means the Brownian motion getting greater. Comparing the graph with $n = 50$ and the one with $n = 2000$, for both Arithmetic Brownian Motion (ABM) and Geometric Brownian Motion (GBM), we get that the graphs with $n = 50$ are smoother than the ones with $n = 2000$. The small value of $dt = \frac{T}{n}$ improve the quality of the appropriate sample path to a true Brownian motion.

Trend of ABM path with $n = 50$ is not consistent with ABM path with $n = 2000$. The trend of ABM path with $n = 50$ fluctuates, with tendency of downtrend at the end of time. Meanwhile, the trend of ABM path with $n = 2000$ has tendency to uptrend from beginning to the end of time. It is consistent with GBM path, both with $n = 50$ and with $n = 2000$. It also can be seen that ABM values, in special occasion, can be under the initial value. It implies if we set \$1 as the initial value, the movement of the price can be negative at some points. It makes ABM becoming not reliable to model any underlying asset price because the price will never below zero.

The code in R:

```
RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function Addins
ABM and GBM simulation.R x GBMchange.R x Untitled1* x Final ABM and GBM simulation.R x
Source on Save Run Source
1 #*****
2 # ABM simulation
3 #*****
4
5 #input parameter values
6 tau <- 1 #time horizon (year)
7 mu <- 0.05 #drift
8 sigma <- 0.15 #volatility
9 X0 <- 10 #initial value
10
11 #Simulate ABM Path with n = 50
12 N <- 50
13 dt <- tau/N
14 time <- seq(from=0, to=tau, by=dt)
15 length(time)
16
17 Z <- rnorm(N, mean = 0, sd = 1)
18 dw <- Z*sqrt(dt)
19 w <- c(0, cumsum(dw))
20
21 X_ABM <- numeric(N+1)
22 X_ABM[1] <- X0
23 for(i in 2:length(X_ABM)){
24   X_ABM[i] <- X_ABM[i-1] + mu*dt + sigma*dw[i-1]
25 }
26
27 #Plot against time t
28 plot(time, X_ABM, type = "l", col = "blue", main = "ABM Path with n = 50",
29       xlab = expression("t"[i]), ylab = expression("w"[t[i]]))
30
31 #Simulate ABM Path with n = 2000
32 N1 <- 2000
33 dt1 <- tau/N1
34 time1 <- seq(from=0, to=tau, by=dt1)
35 length(time1)
36
37 Z1 <- rnorm(N1, mean = 0, sd = 1)
38 dw1 <- Z1*sqrt(dt1)
39 w1 <- c(0, cumsum(dw1))
40
41 X_ABM1 <- numeric(N1+1)
42 X_ABM1[1] <- X0
43 for(i in 2:length(X_ABM1)){
44   X_ABM1[i] <- X_ABM1[i-1] + mu*dt1 + sigma*dw1[i-1]
45 }
46
47 #Plot against time t
48 plot(time1, X_ABM1, type = "l", col = "darkgreen", main = "ABM Path with n = 2000",
49       xlab = expression("t"[i]), ylab = expression("w"[t[i]]))
50
51 -
52 Z <- rnorm(n, mean = 0, sd = 1)
53 dw <- Z*sqrt(dt)
54 w <- c(0, cumsum(dw))
55
56 X_GBM <- numeric(n+1)
57 X_GBM[1] <- X0
58 for(i in 2:length(X_GBM)){
59   X_GBM[i] <- X_GBM[i-1] + mu*X_GBM[i-1]*dt + sigma*X_GBM[i-1]*dw[i-1]
60 }
61
62 plot(time, X_GBM, type = "l", col = "blue", main = "GBM path with n = 50",
63       xlab = expression("t"[i]), ylab = expression("w"[t[i]]))
64
65 #Simulate GBM Path with n = 2000
66 n1 <- 2000
67 dt1 <- tau/n1
68 time1 <- seq(from=0, to=tau, by=dt1)
69 length(time1)
70
71 Z1 <- rnorm(n1, mean = 0, sd = 1)
72 dw1 <- Z1*sqrt(dt1)
73 w1 <- c(0, cumsum(dw1))
74
75 X_GBM1 <- numeric(n1+1)
76 X_GBM1[1] <- X0
77 for(i in 2:length(X_GBM1)){
78   X_GBM1[i] <- X_GBM1[i-1] + mu*X_GBM1[i-1]*dt1 + sigma*X_GBM1[i-1]*dw1[i-1]
79 }
80
81 plot(time1, X_GBM1, type = "l", col = "darkgreen", main = "GBM path with n = 2000",
82       xlab = expression("t"[i]), ylab = expression("w"[t[i]]))
83
```



Comment: Both ABM and GBM models can capture the increasing trend of the real stock price, as I set $n = 1000$. However, if we look at the value of W_t , GBM model is better in capturing the movement of the price compared to ABM. It can be seen that the value of real price starts

at \$80, so does with GBM model. On the other hand, ABM model starts at \$77.4, which is lower than the real data.

The code in R:

```
RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function Addins
Final ABM and GBM simulation.R x Untitled2* x Untitled3* x AAPLclose x AAPL x
Source on Save Run Addins
1 #Importing the required library
2 library(quantmod)
3 library(timesSeries)
4 library(xts)
5 #Importing monthly data from yahoo.finance
6 getSymbols("AAPL", return.class = 'xts', index.class = 'date', from="2020-01-01", to="2021-12-31", periodicity = "monthly")
7 AAPLclose = AAPL[, 4]
8 plot(AAPLclose, main = "Closing Price of AAPL", ylab = "Stock Price")
9
10 AAPL.ts = ts(AAPLclose) #changing into time series
11
12 #Creating Drift Function and calculating the drift of the time series
13 drift.f = function(S, lag=1){
14   N = length(S)
15   if (N < 1 + lag){
16     stop("S must be greater than 2 + lag")
17   }
18   ct = S[(1+lag):N]
19   pt = S[1:(N-lag)]
20   t = 1
21   dt = t/N
22   stk.R = (ct-pt)/pt
23   mu.hat = sum(stk.R)/(N*dt)
24   mu.hat
25 }
26
27 drift.f(AAPL.ts)
28 drift = drift.f(AAPL.ts)
29
30 ##creating Diffusion Function and calculating the diffusion of the time series
31 vol.f = function(S, lag=1){
32   N = length(S)
33   if (N < 1 + lag){
34     stop("S must be greater than 2 + lag")
35   }
36   ct = S[(1+lag):N]
37   pt = S[1:(N-lag)]
38   Diff = ct - pt
39   tt = 1
40   dt = tt/N
41   stk.R = (ct-pt)/pt
42   mu.hat = mean(stk.R)
43   hat.sig2 = sum((stk.R-mu.hat)^2)/((N-1)*dt)
44   hat.sig = sqrt(hat.sig2)
45   hat.sig
46 }
47
48 vol.f(AAPL.ts)
49 diffusion = vol.f(AAPL.ts)
50
51 #Input and calculating Parameters
52 Nsim = 1000 #numbers of simulation
53 dT <- 1/Nsim #time horizon = 1
54 Time <- seq(from=0, to=1, by=dT)
55 length(Time)
56 X0 <- AAPL.ts[1] #initial value for simulation
57 Z <- rnorm(Nsim, mean = 0, sd = 1)
58 dw <- Z*sqrt(dT)
59 w <- c(0, cumsum(dw))
60
61 #ABM Simulation
62 X_ABM <- numeric(Nsim+1)
63 X_ABM[1] <- X0
64 for(i in 2:length(X_ABM)){
65   X_ABM[i] <- X_ABM[i-1] + drift*dT + diffusion*dw[i-1]
66 }
67
68 plot(Time, X_ABM, type = "l", col = "darkgreen", main = "ABM Modeling of AAPL Closing Price",
69       xlab = expression("t"[i]), ylab = expression("w"[t[i]]))
70
71 #GBM Simulation
72 X_GBM <- numeric(Nsim+1)
73 X_GBM[1] <- X0
74 for(i in 2:length(X_GBM)){
75   X_GBM[i] <- X_GBM[i-1] + drift*X_GBM[i-1]*dT + diffusion*X_GBM[i-1]*dw[i-1]
76 }
77
78 plot(Time, X_GBM, type = "l", col = "blue", main = "GBM Modeling of AAPL Closing Price",
79       xlab = expression("t"[i]), ylab = expression("w"[t[i]]))
80
81
61:13 (Top Level) R Script
Console
```