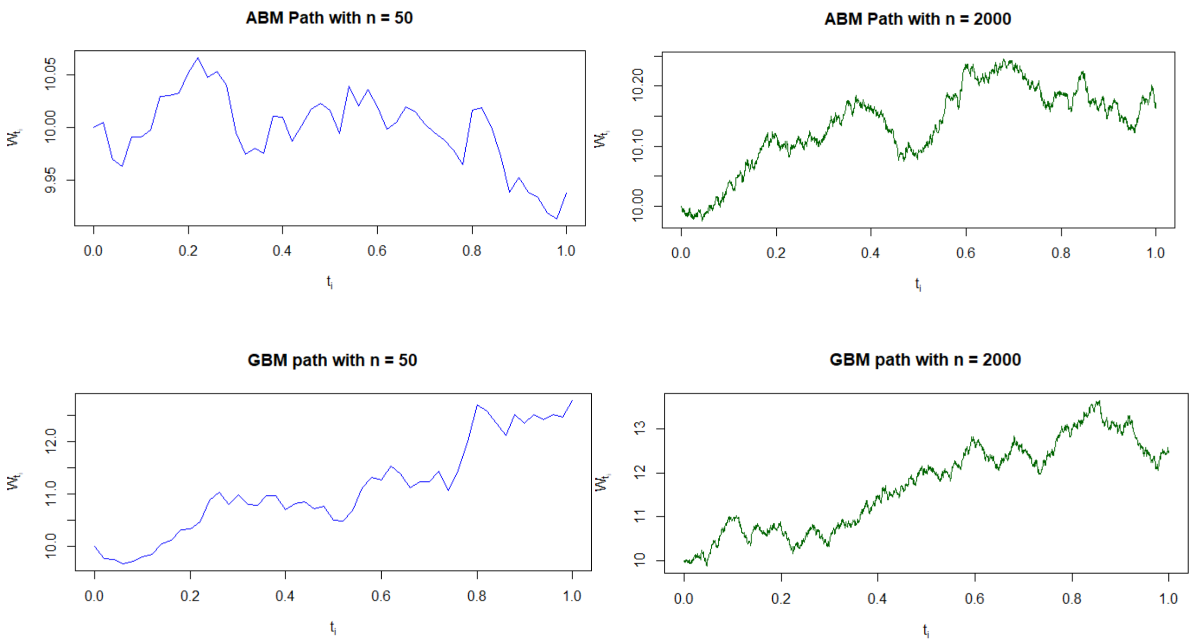
PROJECT 2 REPORT



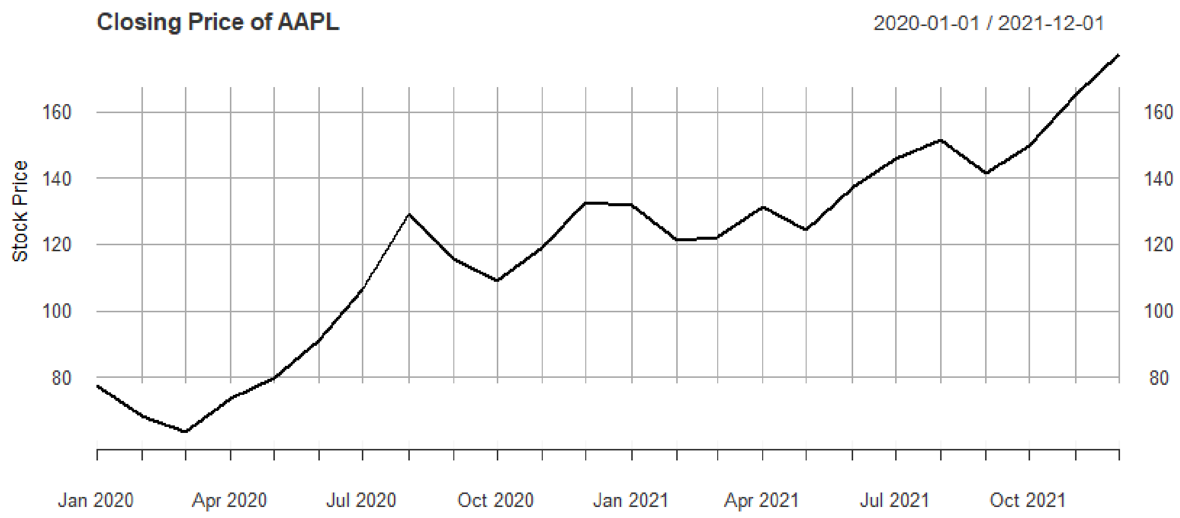
Comment:

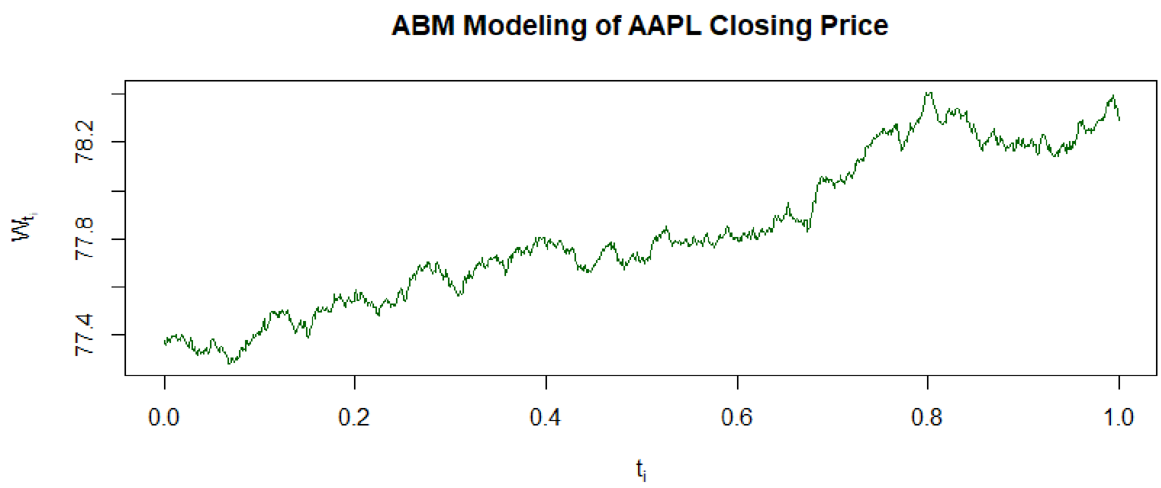
The higher the value of n, the smaller the value of time change $dt = \frac{T}{n}$. It means the Brownian motion getting greater. Comparing the graph with n = 50 and the one with n = 2000, for both Arithmetic Brownian Motion (ABM) and Geometric Brownian Motion (GBM), we get that the graphs with n = 50 are smother 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.

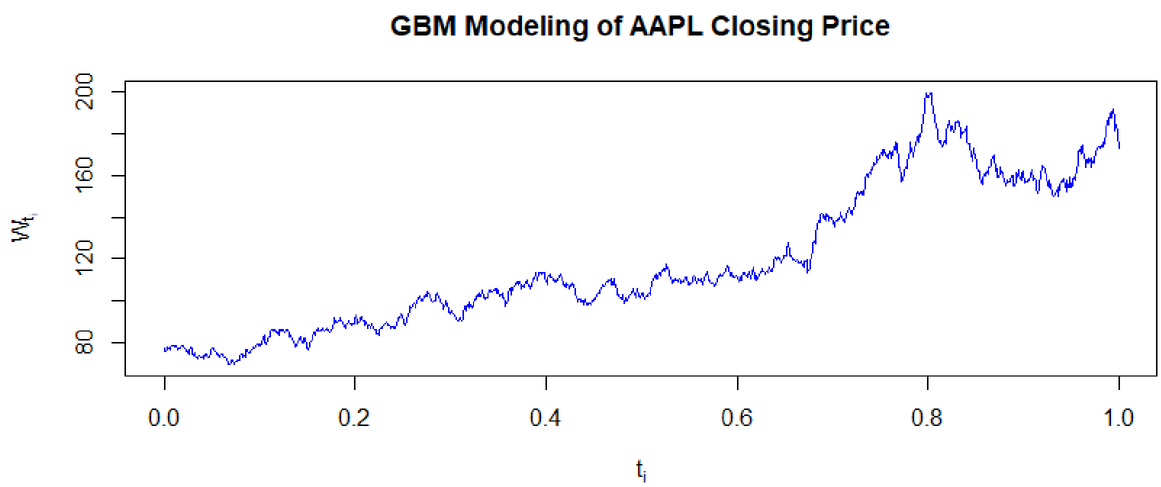
Trend of ABM path with n = 50 is not consistent with ABM path with n = 2000. The trend of ABM path with n = 50 fluctuates, with tendency of downtrend at the end of time. Meanwhile, the trend of ABM path with n = 2000 has tendency to uptrend from beginning to the end of time. It is consistent with GBM 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:

```r
#*********************
#   ABM simulation
#*********************

#input parameter values
tau <- 1 #time horizon (year)
mu <- 0.05 #drift
sigma <- 0.15 #volatility
X0 <- 10 #initial value

#Simulate ABM Path with n = 50
N <- 50
dt <- tau/N
time <- seq(from=0, to=tau, by=dt)
length(time)

Z <- rnorm(N, mean = 0, sd = 1)
dW <- Z*sqrt(dt)
W <- c(0, cumsum(dW))

X_ABM <- numeric(N+1)
X_ABM[1] <- X0
for(i in 2:length(X_ABM)){
  X_ABM[i] <- X_ABM[i-1] + mu*dt + sigma*dW[i-1]
}

#Plot against time t
plot(time, X_ABM, type = "l", col = "blue", main = "ABM Path with n = 50",
     xlab = expression("t"[i]), ylab = expression("W"[t[i]]))

#Simulate ABM Path with n = 2000
N1 <- 2000
dt1 <- tau/N1
time1 <- seq(from=0, to=tau, by=dt1)
length(time1)

Z1 <- rnorm(N1, mean = 0, sd = 1)
dW1 <- Z1*sqrt(dt1)
W1 <- c(0, cumsum(dW1))

X_ABM1 <- numeric(N1+1)
X_ABM1[1] <- X0
for(i in 2:length(X_ABM1)){
  X_ABM1[i] <- X_ABM1[i-1] + mu*dt1 + sigma*dW1[i-1]
}

#Plot against time t
plot(time1, X_ABM1, type = "l", col = "darkgreen", main = "ABM Path with n = 2000",
     xlab = expression("t"[i]), ylab = expression("W"[t[i]]))


z <- rnorm(n, mean = 0, sd = 1)
dw <- z*sqrt(dT)
w <- c(0, cumsum(dw))

X_GBM <- numeric(n+1)
X_GBM[1] <- X0
for(i in 2:length(X_GBM)){
  X_GBM[i] <- X_GBM[i-1] + mu*X_GBM[i-1]*dT + sigma*X_GBM[i-1]*dw[i-1]
}

plot(Time, X_GBM, type = "l", col = "blue", main = "GBM path with n = 50",
     xlab = expression("t"[i]), ylab = expression("W"[t[i]]))

#Simulate GBM Path with n = 2000
n1 <- 2000
dT1 <- tau/n1
Time1 <- seq(from=0, to=tau, by=dT1)
length(Time1)

z1 <- rnorm(n1, mean = 0, sd = 1)
dw1 <- z1*sqrt(dT1)
w1 <- c(0, cumsum(dw1))

X_GBM1 <- numeric(n1+1)
X_GBM1[1] <- X0
for(i in 2:length(X_GBM1)){
  X_GBM1[i] <- X_GBM1[i-1] + mu*X_GBM1[i-1]*dT1 + sigma*X_GBM1[i-1]*dw1[i-1]
}

plot(Time1, X_GBM1, type = "l", col = "darkgreen", main = "GBM path with n = 2000",
     xlab = expression("t"[i]), ylab = expression("W"[t[i]]))
```







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:

```r
#Importing the required library
library(quantmod)
library(timeSeries)
library(xts)
#Impoting monthly data from yahoo.finance
getSymbols("AAPL", return.class = 'xts', index.class = 'Date', from="2020-01-01", to="2021-12-31", periodicity = "monthly")
AAPLclose = AAPL[ , 4]
plot(AAPLclose, main = "Closing Price of AAPL", ylab = "Stock Price")

AAPL.ts = ts(AAPLclose) #changing into time series

#Creating Drift Function and calculating the drift of the time series
drift.f = function(S, lag=1){
  N = length(S)
  if (N < 1 + lag){
    stop("S must be greater than 2 + lag")
  }
  ct = S[(1+lag):N]
  pt = S[1:(N-lag)]
  t = 1
  dt = t/N
  stk.R =(ct-pt)/pt
  mu.hat = sum(stk.R)/(N*dt)
  mu.hat
}

drift.f(AAPL.ts)
drift = drift.f(AAPL.ts)

##Creating Diffusion Function and calculating the diffusion of the time series
vol.f = function (S, lag=1){
  N = length(S)
  if (N < 1 + lag){
    stop("S must be greater than 2 + lag")
  }
  ct = S[(1+lag):N]
  pt = S[1:(N-lag)]
  Diff = ct - pt
  tt = 1
  dt = tt/N
  stk.R = (ct-pt)/pt
  mu.hat = mean(stk.R)
  hat.sig2 = sum((stk.R-mu.hat)^2)/((N-1)*dt)
  hat.sig = sqrt(hat.sig2)
  hat.sig
}

vol.f(AAPL.ts)
diffusion = vol.f(AAPL.ts)

#Input and calculating Parameters
Nsim = 1000 #numbers of simulation
dT <- 1/Nsim #time horizon = 1
Time <- seq(from=0, to=1, by=dT)
length(Time)
X0 <- AAPL.ts[1] #initial value for simulation
z <- rnorm(Nsim, mean = 0, sd = 1)
dw <- z*sqrt(dT)
w <- c(0, cumsum(dw))

#ABM Simulation
X_ABM <- numeric(Nsim+1)
X_ABM[1] <- X0
for(i in 2:length(X_ABM)){
  X_ABM[i] <- X_ABM[i-1] + drift*dT + diffusion*dw[i-1]
}

plot(Time, X_ABM, type = "l", col = "darkgreen", main = "ABM Modeling of AAPL Closing Price",
     xlab = expression("t"[i]), ylab = expression("W"[t[i]]))

#GBM Simulation
X_GBM <- numeric(Nsim+1)
X_GBM[1] <- X0
for(i in 2:length(X_GBM)){
  X_GBM[i] <- X_GBM[i-1] + drift*X_GBM[i-1]*dT + diffusion*X_GBM[i-1]*dw[i-1]
}

plot(Time, X_GBM, type = "l", col = "blue", main = "GBM Modeling of AAPL Closing Price",
     xlab = expression("t"[i]), ylab = expression("W"[t[i]]))
```