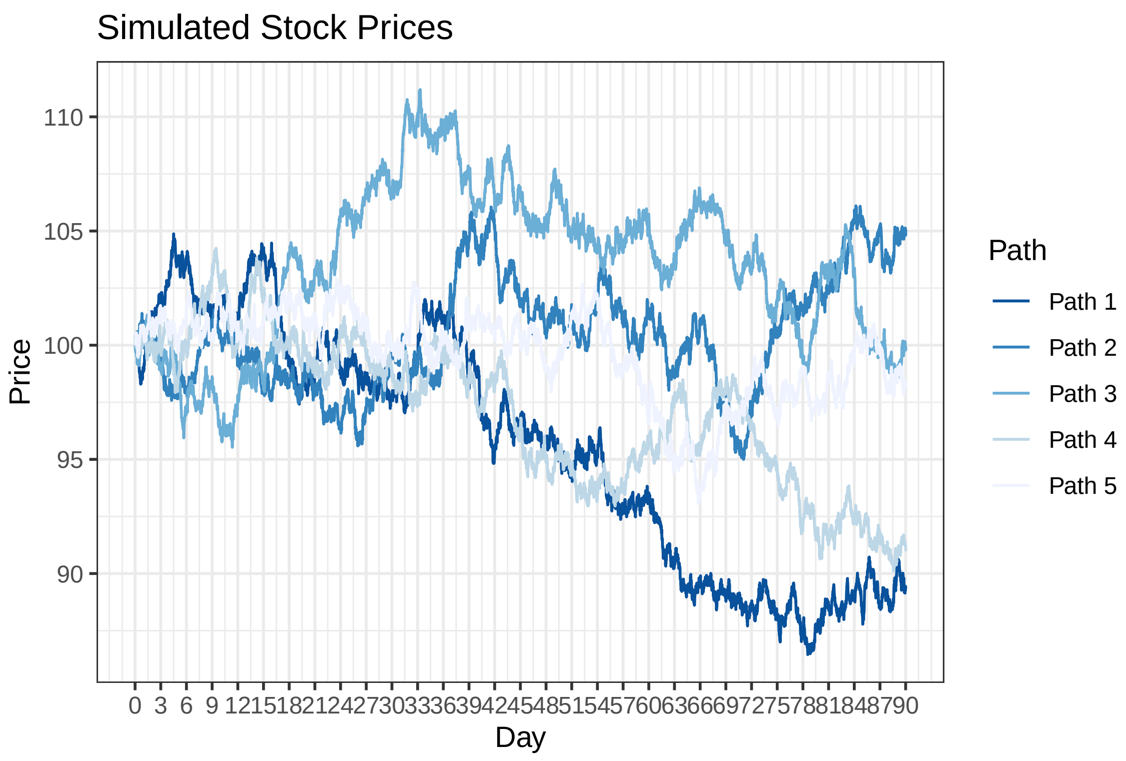
Assignment 2

Sanket Abhyankar, Gourprabh, Tingting Zhou, Niko Hiananto

Question 1. Black-Scholes: Closed Form v. Monte Carlo Simulation

a. Path Simulation

A simulation of 5 stock price paths under the risk-neutral measure is given by the plot below



b. Black-Scholes call option price

The Black-Scholes call option price is

```
##      prc
## 4.614997
```

c. Monte Carlo Price

The confidence intervals of the call option prices using the Monte Carlo simulation for different paths amounts is shown in the table below

Table 1: Confidence Intervals

	Lower	Fit	Upper
100 Paths	2.454	3.514	4.575
1,000 Paths	4.249	4.659	5.069
1,000,000 Paths	4.598	4.611	4.624
100,000,000 Paths	4.614	4.616	4.617

As the number of paths increases, the 95% confidence intervals becomes tighter.

The prices for the call option using the Black-Scholes formula and the Monte Carlo simulation is summarised in the table below

Table 2: Comparison Price

	Call Price
BS Price	4.615
100 Paths	3.514
1,000 Paths	4.659
1,000,000 Paths	4.611
100,000,000 Paths	4.616

Again, as the number of paths increases the simulated call price converges to the actual price from the Black-Scholes formula.

Question 2. Down-And-Out Put Option Price by Monte-Carlo Simulation

a. Simulated Paths

The summary for the simulation is shown by the table below.

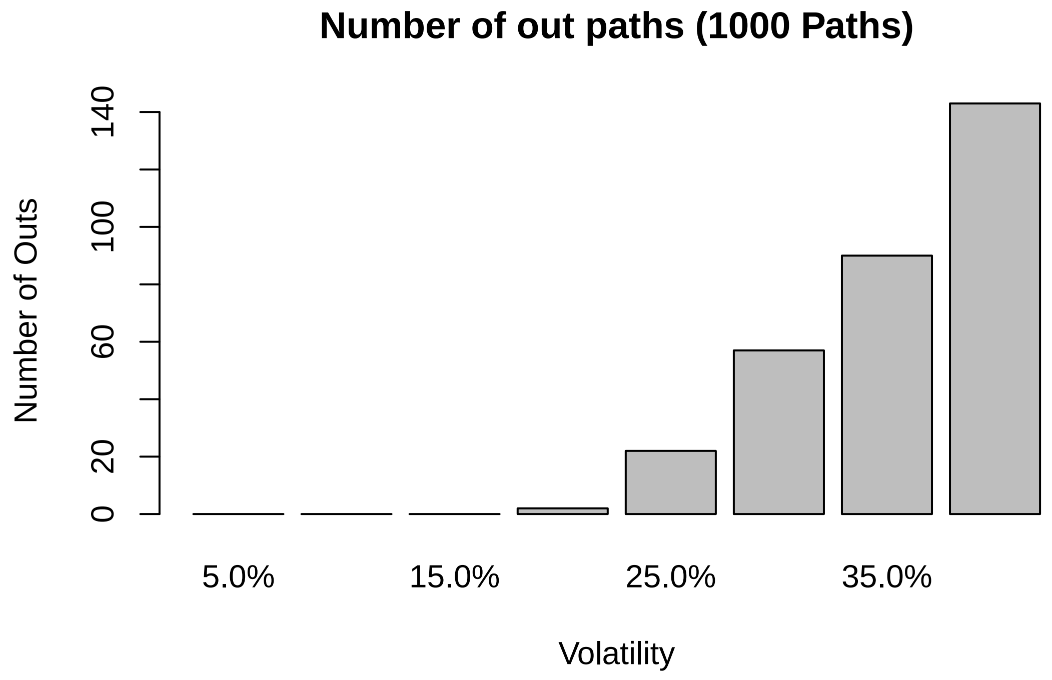
Table 3: Summary: Volatility = .20 (1000 Paths)

Number of Out Paths	2.000
Lower Down&Out	1.322
Fitted Down&Out	1.526
Upper Down&Out	1.729
BS Price	1.534

The number of out paths is quite small as the barrier level is quite far away from the spot price and the volatility of the underlying is only 20%. The fitted price of the put option under the simulation is a little bit higher than the Black-Scholes put option price, however the confidence interval obtained from the simulation is large as well as the number of paths is only 1,000. The down and out put option price is expected to have lower price than a vanilla BS put option price since the down and out option will be worthless if the underlying crosses the barrier, whereas a vanilla put option price will keep gaining value as the underlying price decreases.

b. Down-and-Out under different volatilities

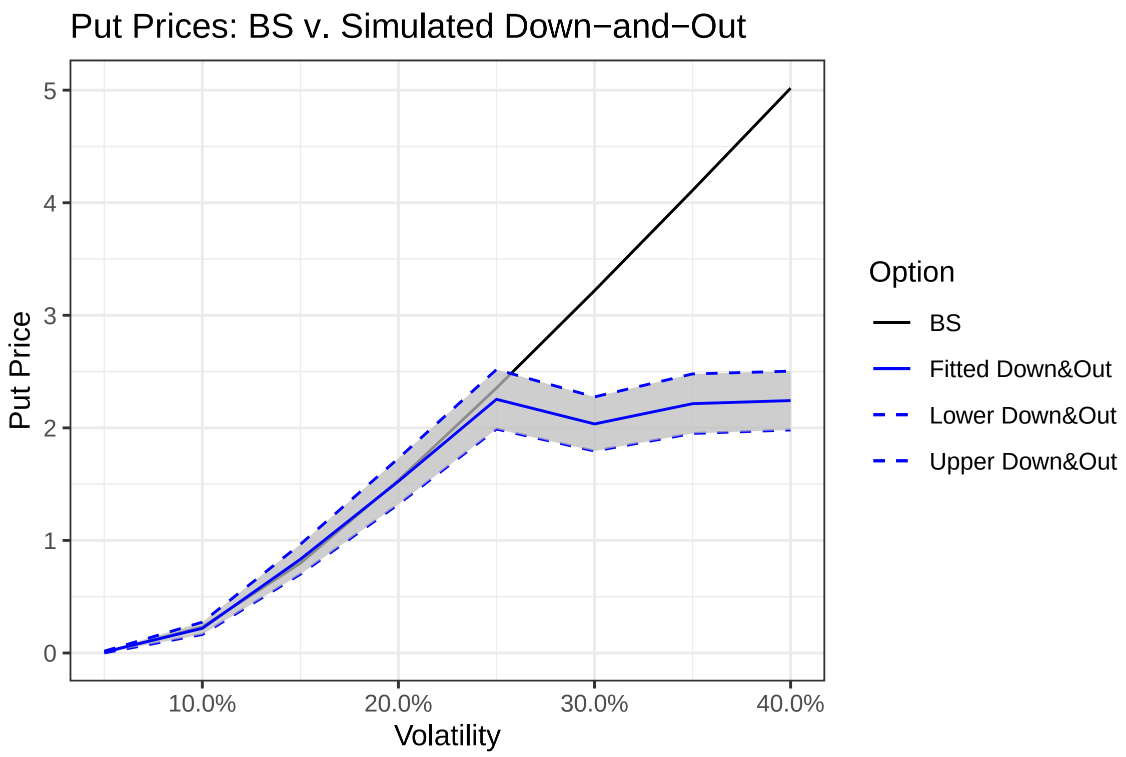
The number of “out” paths under the different volatilities can be seen by the barplot below



As expected, the higher the volatility of the underlying is, the greater the probability of the underlying crossing the barrier and hence the option will be worthless and the number of “out” paths will be greater. In low volatility environments, the probability that the underlying will cross the barrier is small and thus the price of the down and out put option is expected to be closer to the vanilla put option.

c. BS Vanilla v. Down-and-Out Put

The value of the BS vanilla put option and simulated down-and-out put option under different volatilities is shown by the figure below.

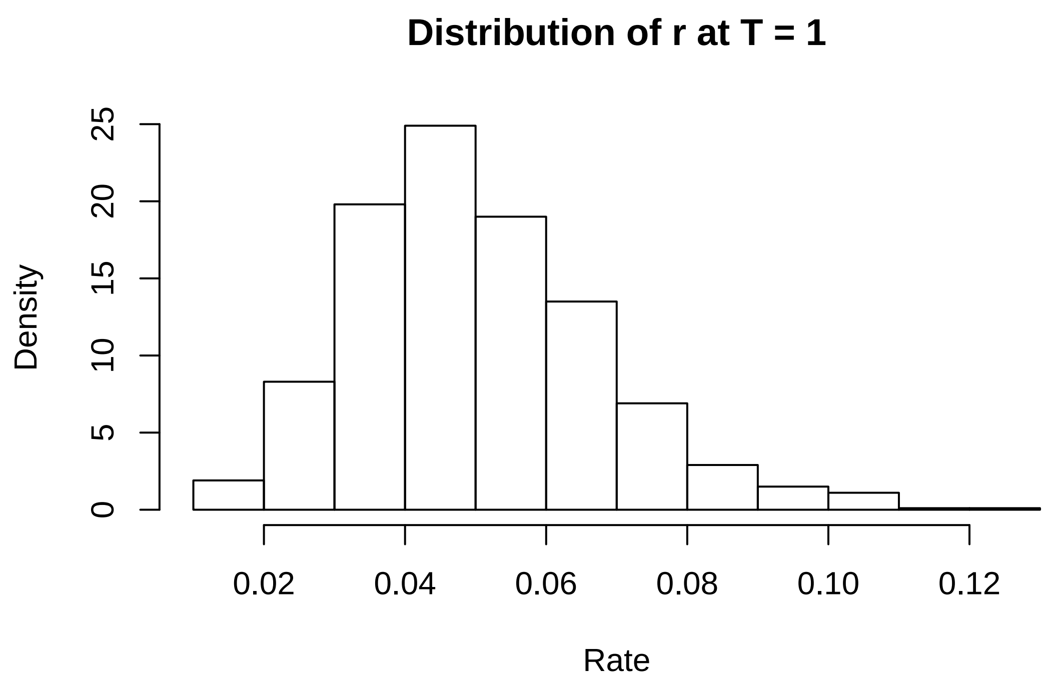


The value of a vanilla put option increases the the volatility of the underlying increases as the greater the volatility is the greater the probability that the option will expire in the money. This is in line with the figure above as the BS vanilla put option price (shown by the black line) increases as the volatility of the underlying increases. However, volatility has two opposing effects on the value of the down-and-out put option. Volatility will still increase the value of the down-and-out put option up to a certain threshold, as seen from the figure above the value of the down-and-out put option keeps increasing until about 30% volatility. At much higher volatility, the probability of the underlying price crossing the barrier would be much higher, which would render the option value worthless - this effectively counteracts the value gained from the increasing probability of the put option expiring in the money. Hence, at some point the volatility would decrease the value of the down-and-out put option as seen from the figure above, where the value of the down-and-out put option peaks at 30% volatility and starts declining as volatility keeps increasing.

Question 3. Pricing Exotic Options

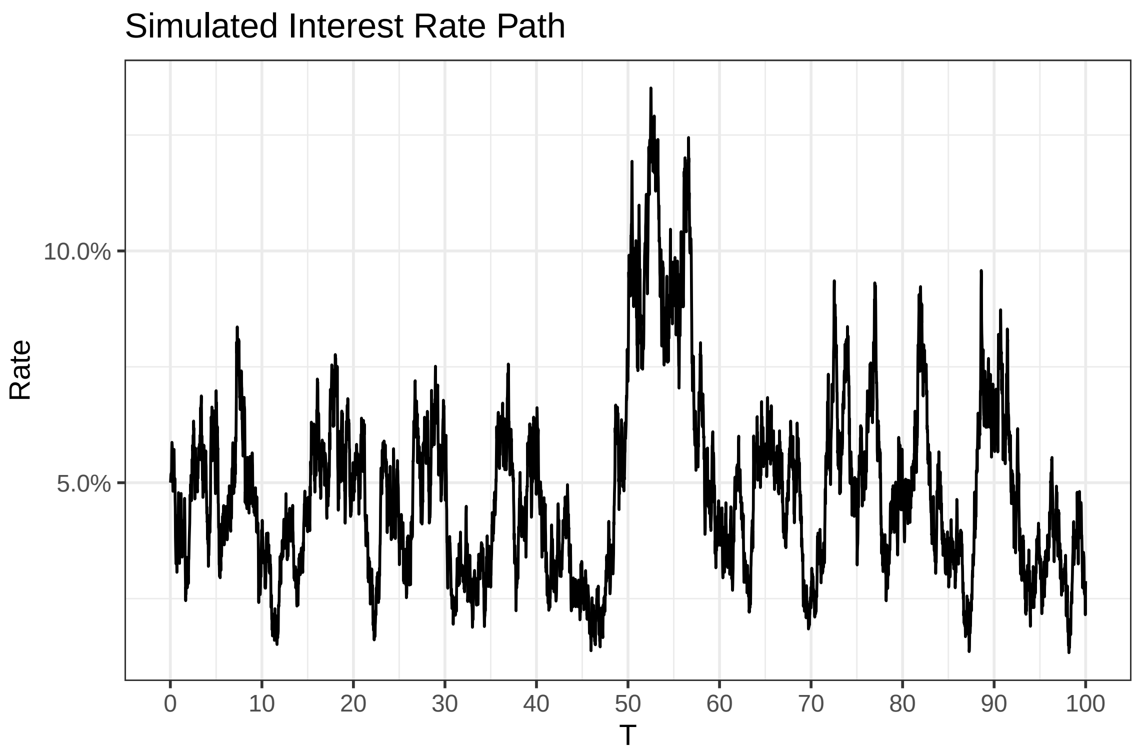
a. Euler Discretization

The distribution of r_T for T = 1 is shown by the histogram below.



b. Interest Rate Trajectory

A simulated interest rate trajectory from t = 0 to T = 100 can be seen by the plot below.



c. Call Option on Asset 1

Using the initial parameters given, the price of the Call option on asset 1 is

```
## [1] 0.7072473
```

d. Monte Carlo Price

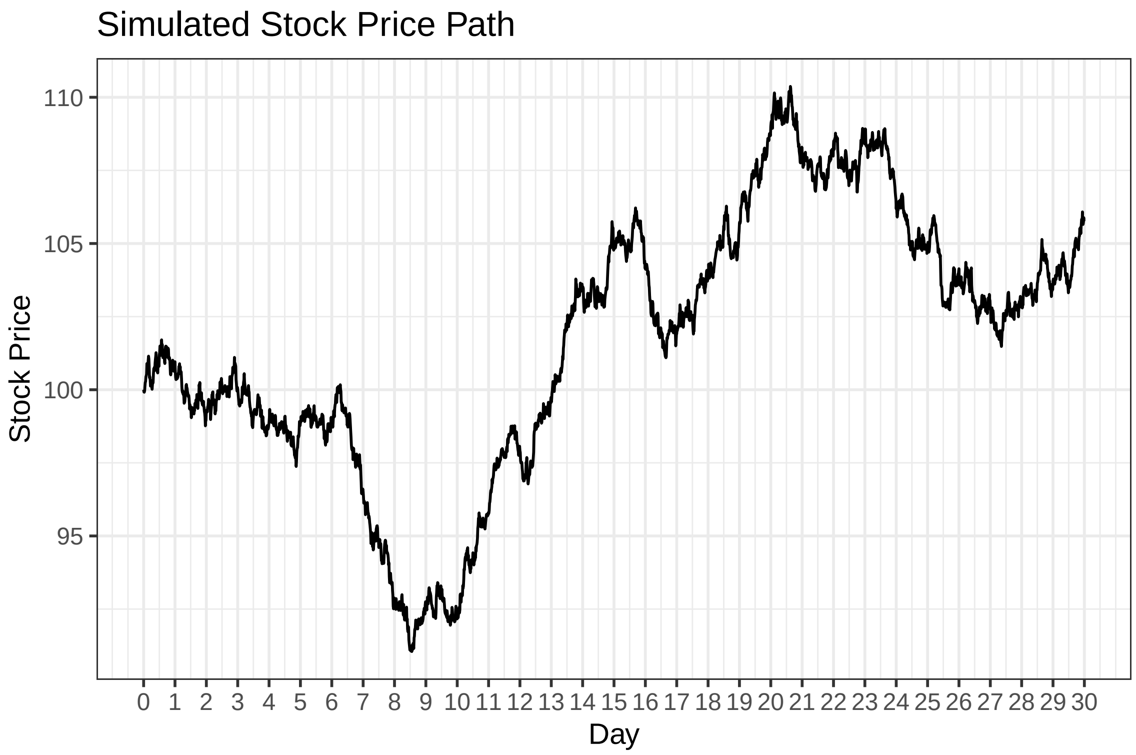
The Monte-Carlo simulation price of the option with the given payoff function is

```
## [1] 1.294339
```

Question 4. Hedging, Large Price Movements, and Transaction Costs

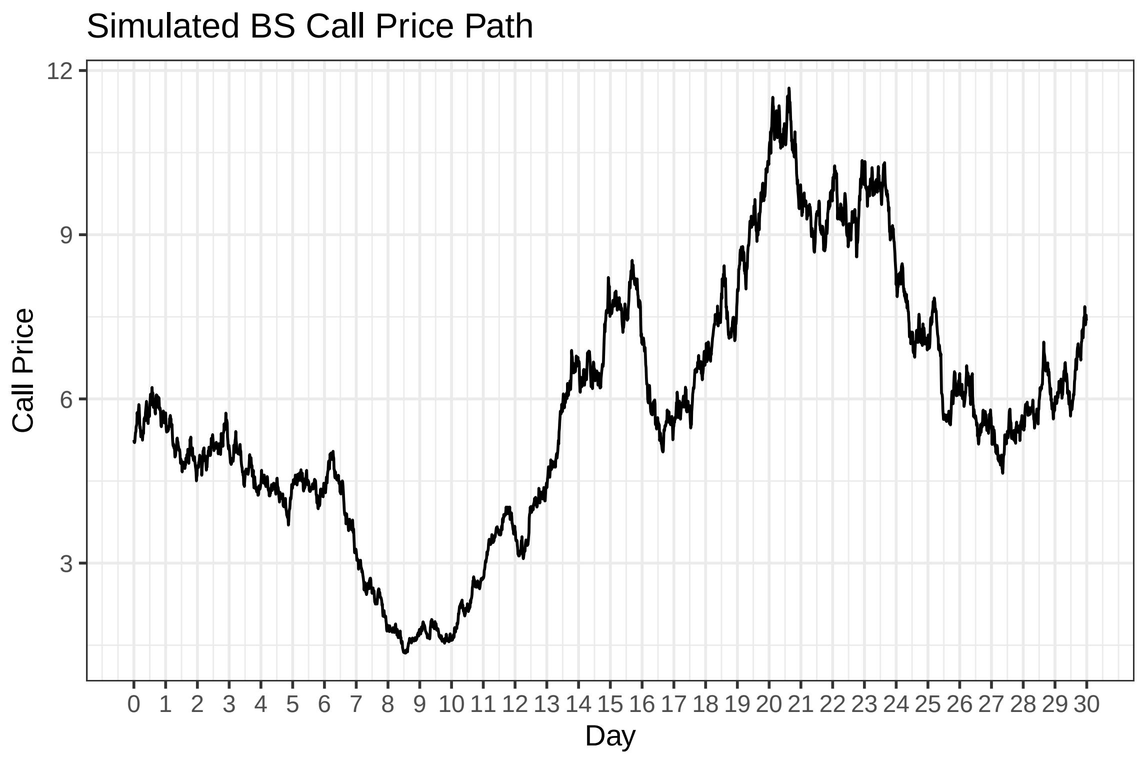
a. Simulated Stock Price

A simulated stock price under the physical probability measure can be seen from the figure below.

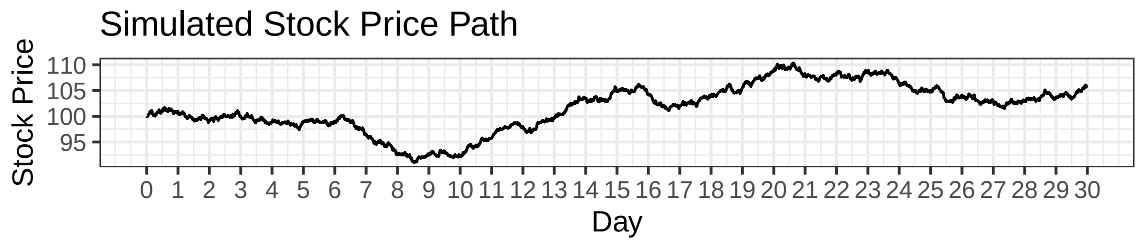


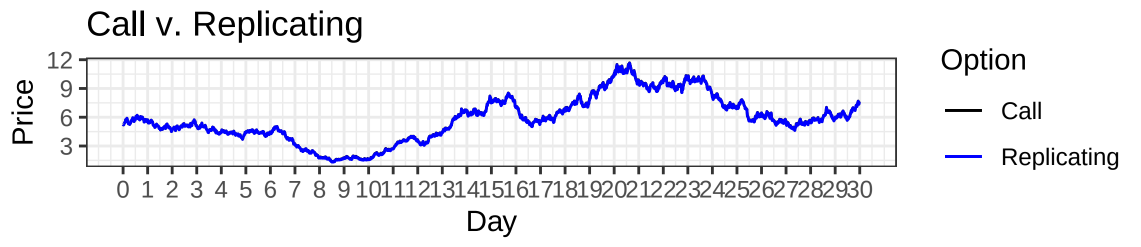
b. Black-Scholes Price

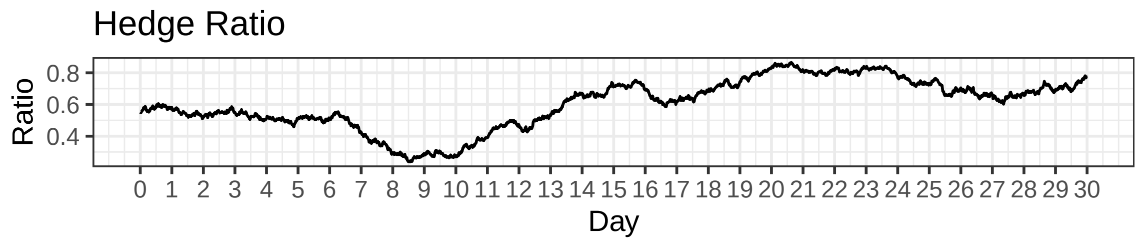
From the simulated stock price from (a), the BS price of the call option along the maturity of the call option is shown by the figure below.



c. Hedging Portfolio

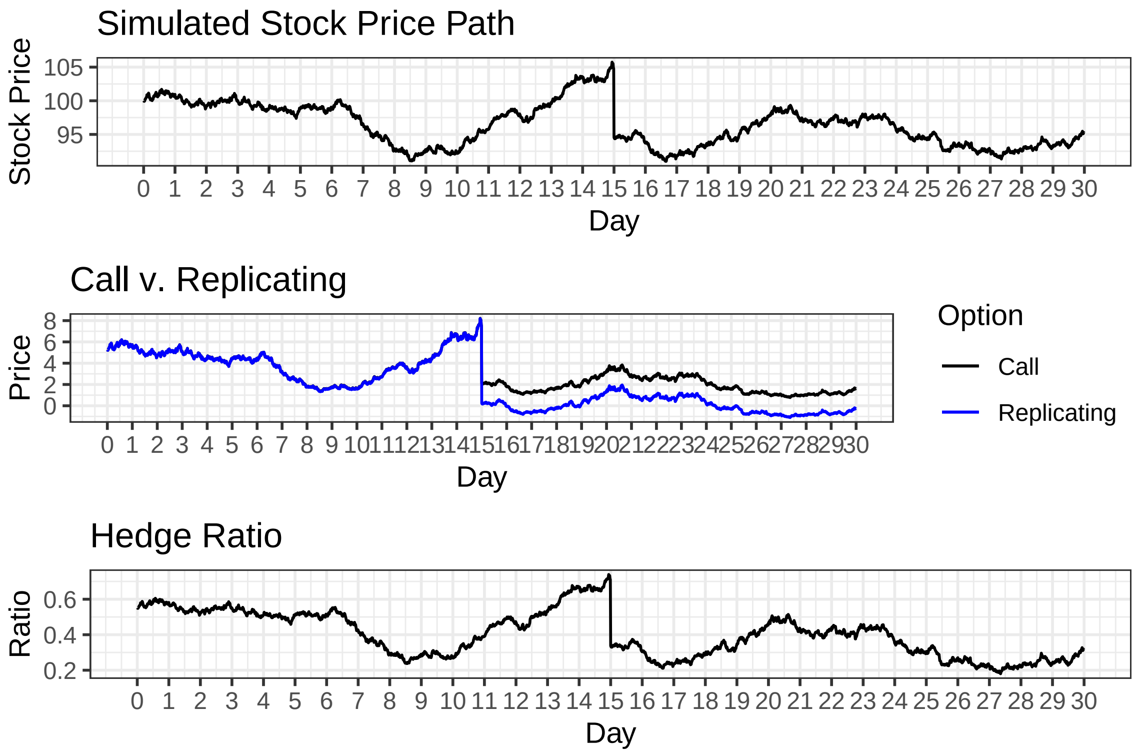






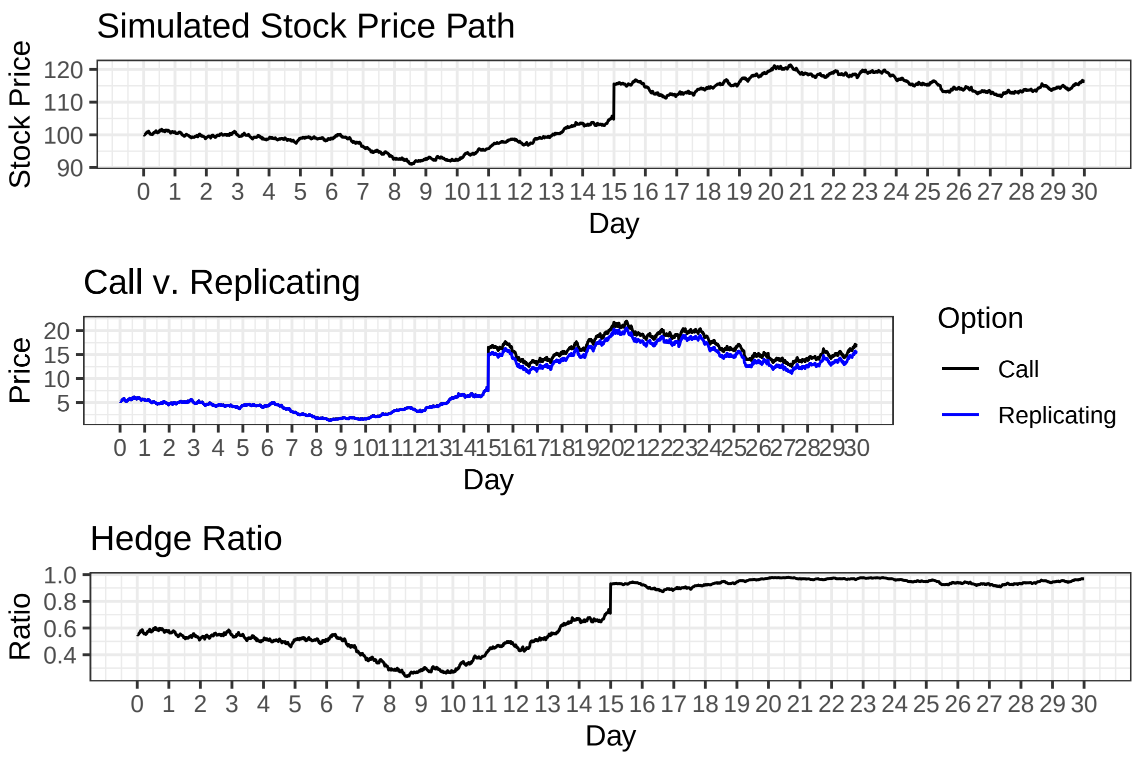
From the figures shown above, the call value and the value of the replicating portfolio matches almost exactly as expected. The hedge ratio decreases as the price of the underlying decreases and increases as the price of the underlying increases. For a call option, the hedge ratio os the delta of the call option and the delta of the call option increases as the option becomes more ITM and decrease as the option becomes more OTM which is shown by the hedge ratio in the above figure.

d. Hedging Portfolio with Downward Jump



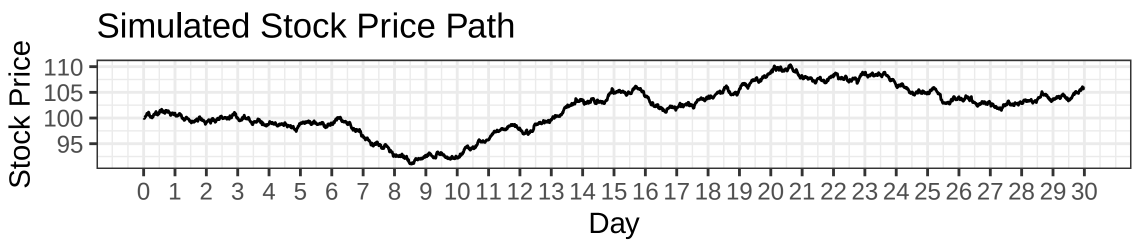
The hedging portfolio matches the call option value closely until the downward jump. The hedge ratio or delta approximates the change in the value of the option for small changes in the underlying price. Due to the large downward jump and since the value of the hedging portfolio depends on the previous hedge ratio and changes in the underlying, the hedge ratio used to create the hedging portfolio cannot account for the jump and hence the value of the hedging portfolio diverges from the actual call value. In other words, the delta of the option before the jump was quite large, and when the jump occurs, the large change in the underlying coupled with the large delta results in a much lower hedging portfolio value (since delta do not approximate the value well for large jumps) than the actual option value as seen from the second figure above.

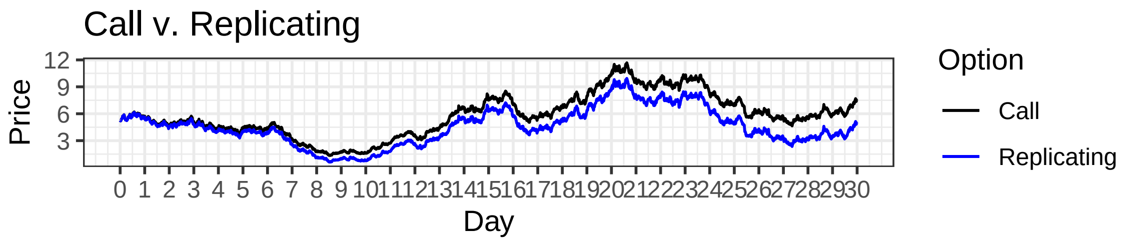
e. Hedging Portfolio with Upward Jump

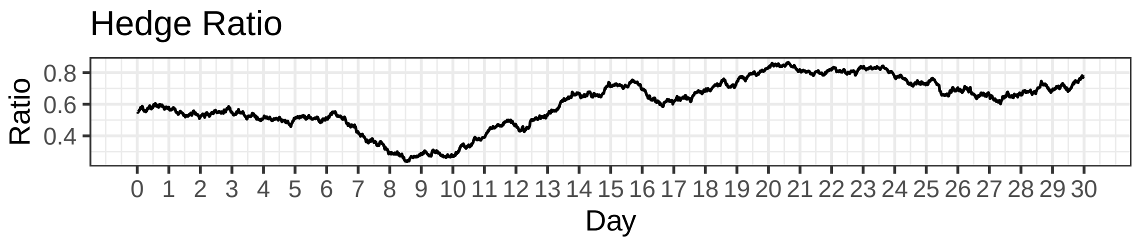


Similarly, as seen in (d), the hedging portfolio value diverges after the upward jump. The reasoning is the same as mentioned before, the hedge ratio simply cannot approximate the value of the portfolio well for large changes in the underlying.

f. Hedging Portfolio with Transaction Cost







With transaction costs, the value of the hedging portfolio decreases over time and diverges from the actual call option value. Transaction costs erode the value of the hedging portfolio as the hedging portfolio rebalances the amount of stock and risk-free asset to match the value of the option. Hence, every time the hedging portfolio rebalances the amount of stock and risk-free asset in the portfolio it will incur transaction costs. Thus, as time goes on, the value of the hedging portfolio keeps decreasing by the transaction cost amount.

Question 5. Heston Model

a. Heston Price v. BS Price

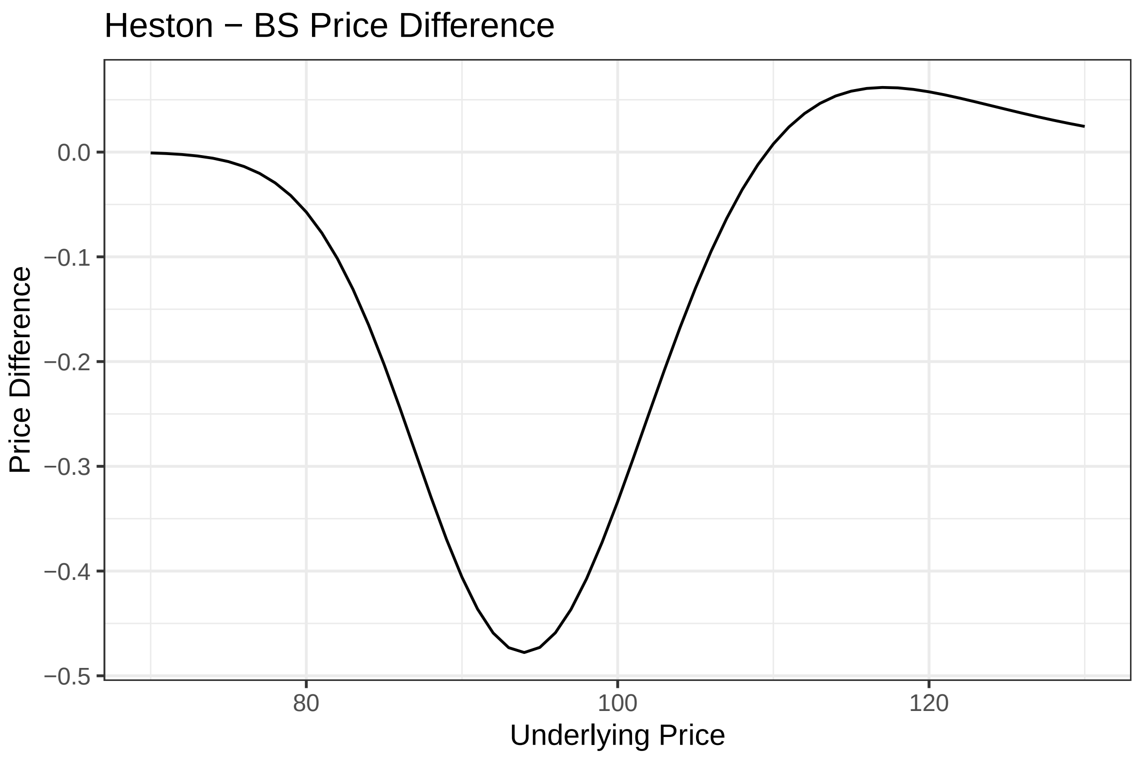
The value of the option under BS and Heston is shown in the table below.

Table 4: Heston v. BS Call Option Price

	Price
Black-Scholes Price	5.017
Heston Price	4.683

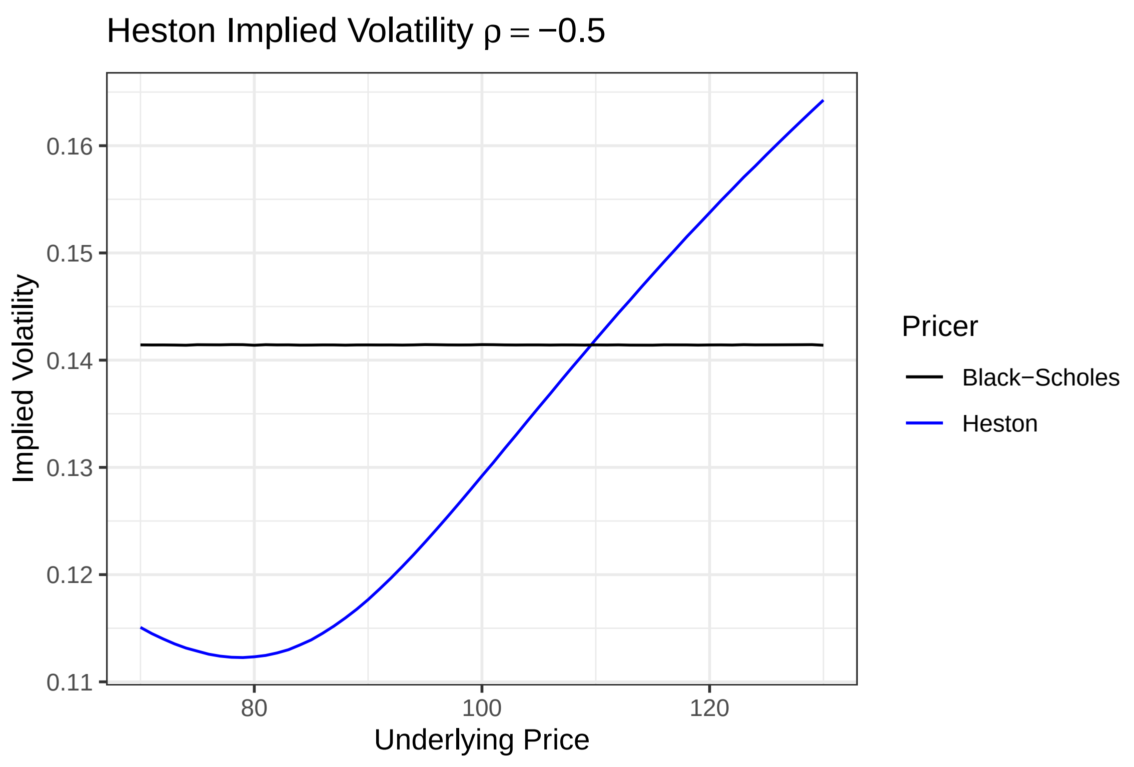
b. Price Difference

The price difference for the Heston call price and the BS call price is shown by figure below.



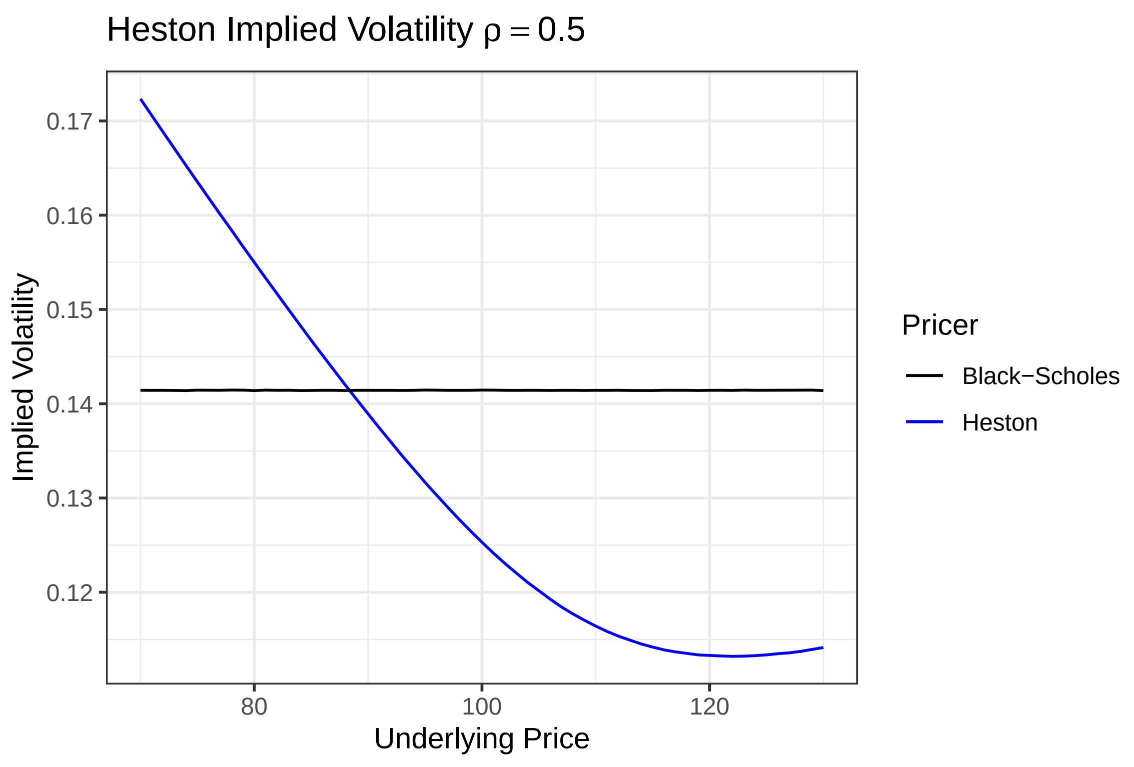
c. Heston Implied Volatility

The implied volatility under the Heston model is shown by the figure below.



The implied volatility is non-constant unlike under the Black-Scholes model. The implied volatility is also increasing as the price of the underlying increases.

d. Heston Implied Volatility



When $\rho = 0.5$, the implied volatility decreases as the underlying price increases, whereas it was increasing as the underlying price increases.

Appendix

```r
knitr::opts_chunk$set(echo = F, fig.height = 4, fig.width = 6, fig.align = "center")
require(tidyverse)
require(kableExtra)
require(purrr)
require(cowplot)
require(latex2exp)
output_kable <- function(table, caption, align = NULL, colnames, rownames = NULL, ...){
  df = table
  rownames(df) = rownames
  df %>% kable(digits = 3, caption = caption, align = align, row.names = T, col.names = colnames, ...)
    kable_styling(latex_options = c("hold_position","striped"), full_width = T)
}
### BS Closed v. Monte Carlo
t = 1/4
k = 100
sigma = 0.2
delta = 0
r = 0.05
n = 96 * 90
h = t/n
s0 = 100

paths = 5
sim = matrix(0, ncol = n+1, nrow = paths)
sim[,1] = s0

#simulate paths
for(i in 2:dim(sim)[2]){
  #assume GBM stock price - path follows S_0 * exp((r - sigma^2)h + sigma * sqrt(h) * Z)
  sim[,i] = sim[,i-1] * exp((r - 0.5 * sigma^2 - delta)*h  + rnorm(dim(sim)[1],0, sigma) * sqrt(h))
}
#a. Plot 5 paths
df_plot = as.data.frame(t(sim))
colnames(df_plot) <- paste0("Path ", 1:5)
df_plot = df_plot %>% mutate(t= row_number())
df_plot = df_plot %>%
 pivot_longer(
   cols = starts_with("Path"),
   names_to = "Path",
   values_to = "prc",)

#plot
ggplot()+
  geom_line(aes(x = t, y = prc, color = Path), data = df_plot) +
  theme_bw() + ylab("Price") + ggtitle("Simulated Stock Prices") + xlab("Day") +
  scale_x_continuous(breaks = seq(0,n,96*3), labels = seq(0,n,96*3)/96) +
  scale_color_brewer(palette="Blues", direction = -1)
#b. BS  Closed Form
bs_formula <- function(s, k, t, delta, sigma, r){
  d1 = 1/(sigma * sqrt(t)) * (log(s/k) + ((r + (sigma^2) / 2 )* t))
  d2 = d1 - (sigma * sqrt(t))
  c = s * exp(-delta * t) * pnorm(d1) - k * pnorm(d2) * exp(- r * t)
  return(c("prc" = c, "delta" = exp(-delta * t) * pnorm(d1), "bond" = c - (s * exp(-delta * t)  * pnorm
}


bs_prc = bs_formula(s0,k,t,delta,sigma,r)[1]
bs_prc
#c. Monte Carlo Price
monte_carlo_prc <- function(s, k, t, delta, sigma, r, paths){
  sim = numeric(paths)
  rand = rnorm(paths,0,1)

  sim = s * exp( (r - delta - 0.5 * sigma^2) * t + (rand * sigma * sqrt(t)))
  payoff = sapply(sim, function(x) max(x - k, 0))
  prc = exp(-r * t) * mean(payoff)
  err = qt(0.975,paths -2) * sd(payoff)/sqrt(paths)
  conf = exp(-r * t) * c(mean(payoff) -  err, mean(payoff), mean(payoff) + err)
  return(list("prc" = prc, "conf" = conf))
}

path_list = c(100,1000,1000000,100000000)
mc = lapply(path_list, function(x) monte_carlo_prc(s0,k,t,delta,sigma,r, x))
mc_prc = unlist(map(mc,1))
mc_conf = t(simplify2array(map(mc,2)))
mc_conf %>% output_kable(caption = "Confidence Intervals", rownames = c("100 Paths", "1,000 Paths", "1,
tbl1 <- matrix(c(bs_prc,mc_prc), ncol = 1)

tbl1 %>% output_kable(caption = "Comparison Price", rownames = c("BS Price", "100 Paths", "1,000 Paths"
                      colnames = "Call Price", align = "c")
### 2
### Down and Out Put Option
t = 1/4
k = 95
barrier = 75
sigma = 0.2
delta = 0
r = 0.05
n = 96 * 90
h = t/n
s0 = 100

paths = 1000

#put price and conf int
monte_carlo_prc_barrier <- function(s, k, h, t, delta, sigma, r, barrier, paths){
  n = t/h
  sim = matrix(0, ncol = n+1, nrow = paths)
  sim[,1] = s

  for(i in 2:dim(sim)[2]){
    sim[, i] = sim[,i-1] * exp((r - -delta - 0.5 * sigma^2)*h  + rnorm(dim(sim)[1],0, sigma) * sqrt(h))
  }
```

```r
  payoff = numeric(paths)
  sim_b = sim * (sim > barrier)
  out = 0
  for(i in 1:paths){
    idx = match(0, sim_b[i,])
    if(!is.na(idx)) {
      sim_b[i, idx:dim(sim_b)[2]] = 0
      out = out + 1
      payoff[i] = 0
    } else{
      payoff[i] = max(k - sim_b[i,dim(sim_b)[2]], 0)
    }

  }


  prc = exp(-r * t) * mean(payoff)
  err = qt(0.975,paths -2) * sd(payoff)/sqrt(paths)
  conf = exp(-r * t) * c(mean(payoff) -  err, mean(payoff), mean(payoff) + err)


  return(list("prc" = prc, "out" = out, "conf" = conf))
}

bs_formula_put <- function(s, k, t, delta, sigma, r){
  d1 = 1/(sigma * sqrt(t)) * (log(s/k) + ((r + (sigma^2) / 2 )* t))
  d2 = d1 - (sigma * sqrt(t))
  p = k * pnorm(-d2) * exp(- r * t) - s * exp(-delta * t) * pnorm(-d1)
  return(c("prc" = p, "delta" = exp(-delta * t) * (pnorm(d1) - 1), "bond" =  k * pnorm(-d2) * exp(- r *
}


sigma_list = c(0.05,0.1,0.15,0.2,.25,0.3,0.35,0.4)

res = lapply(sigma_list, function(x) monte_carlo_prc_barrier(s0,k,h,t,delta,x,r, barrier, paths))

bs_put_prc = sapply(sigma_list, function(x) bs_formula_put(s0,k,t,delta,x,r)[1])
#a
#20% vol
pl_tbl <- matrix(c(res[[4]][[2]], c(res[[4]][[3]]), bs_put_prc[4]), ncol = 1)
pl_tbl %>% output_kable(caption = "Summary: Volatility = .20 (1000 Paths)", align = "c",
                        rownames = c("Number of Out Paths", "Lower Down&Out", "Fitted Down&Out", "Upper
                        colnames = NULL)
#b
#out paths
barplot(unlist(map(res,2)), names.arg = scales::percent(sigma_list), ylab = "Number of Outs", xlab = "V
#c
down_out = simplify2array(map(res,3))

ggplot() +
  geom_line(aes(x = sigma_list, y = bs_put_prc, color = 'BS', linetype = 'BS')) +
  geom_line(aes(x = sigma_list, y = down_out[1,], color = 'Lower Down&Out', linetype = "Lower Down&Out"
  geom_ribbon(aes(sigma_list, ymin=down_out[1,],ymax=down_out[3,]), fill = 'grey', alpha = 0.75) +
  geom_line(aes(x = sigma_list, y = down_out[2,], color = 'Fitted Down&Out' ,linetype = "Fitted Down&Out
  geom_line(aes(x = sigma_list, y = down_out[3,], color = 'Upper Down&Out' , linetype = "Upper Down&Out
  xlab("Volatility") + ylab("Put Price") + ggtitle("Put Prices: BS v. Simulated Down-and-Out") +
  theme_bw() +
  scale_color_manual(name = "Option",
                     values = c("BS" = "black", "Lower Down&Out" = "blue", "Fitted Down&Out" = "blue",
  scale_linetype_manual(name = "Option",
                        values = c("BS" = "solid", "Lower Down&Out" = "dashed", "Fitted Down&Out" = "sol
  scale_x_continuous(labels = scales::percent)
r = 0.05
beta = 0.05
alpha = 0.6
delta = 0.1

#discretize
h = 1/250
t = 1
n = t/h

#a. discretize rate
paths = 1000
r_disc = matrix(0, ncol = n+1, nrow = paths)
r_disc[,1] = r
for(i in 2:dim(r_disc)[2]){
  r_disc[,i] = r_disc[,i-1] + (alpha * (beta - r_disc[,i-1])* h) + delta * sqrt(r_disc[,i-1]) * rnorm(pa
}
#histogram
hist(r_disc[,dim(r_disc)[2]], xlab = "Rate", main = "Distribution of r at T = 1", prob = T)
#b
r = 0.05
delta = 0.1
beta = 0.05
alpha = 0.6
t = 100
h = 1/52
n = t/h
r_sim = numeric(n + 1)
r_sim[1] = r

for(i in 2:length(r_sim)){
  r_sim[i] = r_sim[i-1] + (alpha * (beta - r_sim[i-1])* h) + delta * sqrt(r_sim[i-1]) * rnorm(1,0,1) * s
}

ggplot() +
  geom_line(aes(x = 0:5200, y = r_sim)) + theme_bw() +
  ylab("Rate") + ggtitle("Simulated Interest Rate Path") +
  scale_y_continuous(labels = scales::percent) +
  scale_x_continuous(name = "T", breaks = seq(0,5200,520), labels = seq(0,5200,520)/52)

#c & d
vol11 = 0.1
vol12 = 0.2
vol21 = 0.3
```

```
s1_0 = 10
s2_0 = 10
r = 0.05
beta = 0.05
alpha = 0.6
delta = 0.1
k = 10

t = 0.5
h = 1/250
n = t/h
paths = 10000
r_disc = sim_s1 = sim_s2 = matrix(0,nrow = paths, ncol = n+1)
r_disc[,1] = r
sim_s1[,1] = s1_0
sim_s2[,1] = s2_0

for(i in 2:dim(sim_s1)[2]){

  z = matrix(rnorm(paths*2,0,1), ncol = 2) #B1 & B2
  r_disc[,i] = r_disc[,i-1] + (alpha * (beta - r_disc[,i-1])* h) + delta * sqrt(r_disc[,i-1]) * z[,1] *

  sim_s1[,i] = sim_s1[,i-1] + (sim_s1[,i-1] * r_disc[,i] * h) + (vol11 * sqrt(sim_s1[,i-1]) * z[,1] * s
  sim_s2[,i] = sim_s2[,i-1] + (sim_s2[,i-1] * r_disc[,i] * h) + (vol21 * (sim_s1[,i-1] - sim_s2[,i-1]) *
}
#c
#call option
payoff = sapply(sim_s1[,dim(sim_s1)[2]], function(x) max(x-k,0))
prc = exp(-r * t) * mean(payoff)
prc
#d
#payoff
max_s1_s2 = apply(cbind(apply(sim_s1,1,max), apply(sim_s2,1,max)), 1,max)
payoff = sapply(max_s1_s2, function(x) max(x -k,0))
prc = exp(-r * t) * mean(payoff)
prc
#params
r = 0.05
K = 100
sigma = 0.3
delta = 0
mu = 0.2
s = 100


n = 8 * 12 * 30
t = 30/365
h = t/n

#a simulated stock price path under physical prb
paths = 1
sim = matrix(0, ncol = n+1, nrow = paths)
sim[,1] = s
```

```r
for(i in 2:dim(sim)[2]){
  #assume GBM stock price - path follows S_0 * exp((r - sigma^2)h + sigma * sqrt(h) * Z)
  sim[,i] = sim[,i-1] * exp((mu - 0.5 * sigma^2 - delta)*h  + rnorm(dim(sim)[1],0, sigma) * sqrt(h))
}
#plot underlying
stock_plt <- ggplot() +
  geom_line(aes(x = 0:n, y = sim[1,])) + theme_bw() +
  ylab("Stock Price") + ggtitle("Simulated Stock Price Path") +
  scale_x_continuous(name = "Day", breaks = seq(0,n,96), labels = seq(0,n,96)/96)

stock_plt
#b BS price
maturity = 60/365 - (h * 0:2880)

option_prices = simplify2array(map2(sim[1,], maturity, function(x,y) unname(bs_formula(s = x, k = K, t


#plot
ggplot() +
  geom_line(aes(x = 0:n, y = option_prices)) + theme_bw() +
  ylab("Call Price") + ggtitle("Simulated BS Call Price Path") +
  scale_x_continuous(name = "Day", breaks = seq(0,n,96), labels = seq(0,n,96)/96)
#c hedging portfolio
hedge_pf <- function(sim_path, trx = 0){
  hedging_pf = matrix(0, nrow = 5, ncol = length(sim_path))
  rownames(hedging_pf) <- c("stock","delta","bond","rep","call")
  #stock prices
  hedging_pf[1,] = sim_path
  #delta & bond
  hedging_pf[2:3,1] = bs_formula(s = s, k = K, t = 60/365, delta = delta, sigma = sigma, r = r)[2:3]
  #pf val
  hedging_pf[4,1] = hedging_pf[1,1] * hedging_pf[2,1] + hedging_pf[3,1]

  #actual opt prices
  hedging_pf[5,1] = bs_formula(s = s, k = K, t = 60/365, delta = delta, sigma = sigma, r = r)[1]

  for(j in 2:length(sim_path)){
    bs = bs_formula(hedging_pf[1,j], K, 60/365 - (h * (j-1)), delta, sigma, r)
    #hedge ratio
    hedging_pf[2,j] = bs[2]

    #value
    hedging_pf[4,j] = hedging_pf[1,j] * hedging_pf[2,j-1] + hedging_pf[3,j-1] * exp(r * h) -
      abs(hedging_pf[2,j-1] - hedging_pf[2,j]) * hedging_pf[1,j] * trx #trx cost

    #new bond
    hedging_pf[3,j] = hedging_pf[4,j] - (hedging_pf[2,j] * hedging_pf[1,j])

    #act BS call price
    hedging_pf[5,j] = bs[1]
  }
  return(hedging_pf)
}
```

```r
hedging_pf_1 = hedge_pf(sim[1,])
#plot
call_rep_plt <- ggplot() +
  geom_line(aes(x = 0:n, y = option_prices, color = 'Call')) +
  geom_line(aes(x = 0:n, y = hedging_pf_1[4,], color = 'Replicating')) +
  theme_bw() +
  ylab("Price") + ggtitle("Call v. Replicating") +
  scale_color_manual(name = "Option",
                     values = c("Call" = "black", "Replicating" = "blue"),
                     labels = c("Call","Replicating")) +
  scale_x_continuous(name = "Day", breaks = seq(0,n,96), labels = seq(0,n,96)/96)

hedge_plt <- ggplot() +
  geom_line(aes(x = 0:n, y = hedging_pf_1[2,])) +
  theme_bw() +
  ylab("Ratio") + ggtitle("Hedge Ratio") +
  scale_x_continuous(name = "Day", breaks = seq(0,n,96), labels = seq(0,n,96)/96)

plot_grid(stock_plt, call_rep_plt, hedge_plt, ncol = 1, align = c('h','v'))
#d.
#downward jump 10%
sim_d = matrix(c(sim[,1:(n/2)], sim[,((n/2)+1): dim(sim)[2]] * 0.9), nrow = 1)

hedging_pf_2 = hedge_pf(sim_d[1,])


#plot underlying
stock_plt <- ggplot() +
  geom_line(aes(x = 0:n, y = sim_d[1,])) + theme_bw() +
  ylab("Stock Price") + ggtitle("Simulated Stock Price Path") +
  scale_x_continuous(name = "Day", breaks = seq(0,n,96), labels = seq(0,n,96)/96)


call_rep_plt <- ggplot() +
  geom_line(aes(x = 0:n, y = hedging_pf_2[5,], color = 'Call')) +
  geom_line(aes(x = 0:n, y = hedging_pf_2[4,], color = 'Replicating')) +
  theme_bw() +
  ylab("Price") + ggtitle("Call v. Replicating") +
  scale_color_manual(name = "Option",
                     values = c("Call" = "black", "Replicating" = "blue"),
                     labels = c("Call","Replicating")) +
  scale_x_continuous(name = "Day", breaks = seq(0,n,96), labels = seq(0,n,96)/96)

hedge_plt <- ggplot() +
  geom_line(aes(x = 0:n, y = hedging_pf_2[2,])) +
  theme_bw() +
  ylab("Ratio") + ggtitle("Hedge Ratio") +
  scale_x_continuous(name = "Day", breaks = seq(0,n,96), labels = seq(0,n,96)/96)


plot_grid(stock_plt, call_rep_plt, hedge_plt, ncol = 1, align = c('h','v'))
#upward jump
sim_u = matrix(c(sim[,1:(n/2)], sim[,((n/2)+1): dim(sim)[2]] * 1.1), nrow = 1)
```

```r
hedging_pf_3 = hedge_pf(sim_u[1,])


#plot underlying
stock_plt <- ggplot() +
  geom_line(aes(x = 0:n, y = sim_u[1,])) + theme_bw() +
  ylab("Stock Price") + ggtitle("Simulated Stock Price Path") +
  scale_x_continuous(name = "Day", breaks = seq(0,n,96), labels = seq(0,n,96)/96)


call_rep_plt <- ggplot() +
  geom_line(aes(x = 0:n, y = hedging_pf_3[5,], color = 'Call')) +
  geom_line(aes(x = 0:n, y = hedging_pf_3[4,], color = 'Replicating')) +
  theme_bw() +
  ylab("Price") + ggtitle("Call v. Replicating") +
  scale_color_manual(name = "Option",
                     values = c("Call" = "black", "Replicating" = "blue"),
                     labels = c("Call","Replicating")) +
  scale_x_continuous(name = "Day", breaks = seq(0,n,96), labels = seq(0,n,96)/96)

hedge_plt <- ggplot() +
  geom_line(aes(x = 0:n, y = hedging_pf_3[2,])) +
  theme_bw() +
  ylab("Ratio") + ggtitle("Hedge Ratio") +
  scale_x_continuous(name = "Day", breaks = seq(0,n,96), labels = seq(0,n,96)/96)


plot_grid(stock_plt, call_rep_plt, hedge_plt, ncol = 1, align = c('h','v'))
#trx cost
hedging_pf_4 = hedge_pf(sim[1,], 0.002)

stock_plt <- ggplot() +
  geom_line(aes(x = 0:n, y = sim[1,])) + theme_bw() +
  ylab("Stock Price") + ggtitle("Simulated Stock Price Path") +
  scale_x_continuous(name = "Day", breaks = seq(0,n,96), labels = seq(0,n,96)/96)


call_rep_plt <- ggplot() +
  geom_line(aes(x = 0:n, y = hedging_pf_4[5,], color = 'Call')) +
  geom_line(aes(x = 0:n, y = hedging_pf_4[4,], color = 'Replicating')) +
  theme_bw() +
  ylab("Price") + ggtitle("Call v. Replicating") +
  scale_color_manual(name = "Option",
                     values = c("Call" = "black", "Replicating" = "blue"),
                     labels = c("Call","Replicating")) +
  scale_x_continuous(name = "Day", breaks = seq(0,n,96), labels = seq(0,n,96)/96)

hedge_plt <- ggplot() +
  geom_line(aes(x = 0:n, y = hedging_pf_4[2,])) +
  theme_bw() +
  ylab("Ratio") + ggtitle("Hedge Ratio") +
  scale_x_continuous(name = "Day", breaks = seq(0,n,96), labels = seq(0,n,96)/96)
```

```r
plot_grid(stock_plt, call_rep_plt, hedge_plt, ncol = 1, align = c('h','v'))
heston_call_prc <- function(s, K, tau, r, q, vt, theta, rho, kappa, sigma){
  #s: current stock price
  #K: strike price
  #tau: time to maturity
  #r: interest rate
  #q: dividend yield
  #vt: current volatility
  #theta: long run average volatility
  #rho: brownian motion correlation
  #kappa: speed of mean reversion
  #sigma: vol


d <- function(phi){
  sqrt( (rho * sigma * i * phi - kappa)^2 + sigma^2 * (i * phi + phi^2))
}

g <- function(phi){
  (kappa - rho * sigma * i * phi - d(phi)) / (kappa - rho * sigma * i * phi + d(phi))
}

A <- function(phi){
  ((r - q) * i * phi * tau) +(kappa * theta / sigma^2) *
    (  (kappa - rho * sigma * i * phi - d(phi)) * tau -
        2 * log( (1 - g(phi) * exp(-d(phi) * tau)) / (1 - g(phi)) )
    )
}

C <- function(phi){
  vt/sigma^2 * (kappa - rho * sigma * i * phi -
    d(phi)) * (1 - exp(- d(phi) * tau))/(1 - g(phi) * exp(-d(phi) * tau))
}

cf <- function(phi){
  exp(A(phi) + C(phi) + i * phi * log(s))
}

P1 <- function(phi) {
  p <- Re(exp(-i * log(K) * phi) * cf(phi - i)/ (i * phi *
      s * exp((r - q) * tau) ) )
  p
}
P2 <- function(phi) {
  p <- Re(exp(-i * log(K) * phi) * cf(phi) / (i * phi) )
  p
}

vP1 <- 0.5 + 1/pi * integrate(P1, lower = 0, upper = Inf)$value
vP2 <- 0.5 + 1/pi * integrate(P2, lower = 0, upper = Inf)$value

res <- s * exp(-q * tau) * vP1 - K * exp(-r * tau) * vP2
```

```r
return(res)
}


#params
i = 0 + 1i
tau = 0.5
vt = 0.01
kappa = 6
theta = 0.02
s = 100
lambda = 0
sigma = 0.3
r = 0.04
rho = -0.5
K = 100
delta = 0

# heston_call_prc(100, 100, 0.5, 0.04, delta, vt, theta, rho, kappa, sigma)

bs_prc = bs_formula(s, K, tau, delta, sqrt(theta), r)

hest_prc = heston_call_prc(s, K, tau, r, delta, vt, theta, rho, kappa, sigma)

rbind(bs_prc[1],hest_prc) %>%  output_kable(colnames = "Price", rownames = c("Black-Scholes Price", "He
stock_prc_list = seq(70,130)

#bs prices
bs_res = sapply(stock_prc_list, function(x) unname(bs_formula(x, K, tau, delta, sqrt(theta), r)[1]))
#heston prices
heston_res = sapply(stock_prc_list, function(x) heston_call_prc(x, K, tau, r, delta, vt, theta, rho, ka


ggplot() +
  geom_line(aes(x = stock_prc_list, y = heston_res - bs_res))+
  theme_bw() + ylab("Price Difference") + xlab("Underlying Price") +
  ggtitle("Heston - BS Price Difference")
#implied volatility function
vol_root <- function(s, k, t, delta, r, prc){
  root_fun <- function(sigma) {
    prc - bs_formula(s, k, t, delta, sigma, r)[1]
  }
  uniroot(root_fun, c(0,1))$root
}

impl_vol_hes <- simplify2array(map2(stock_prc_list, heston_res, function(x,y) vol_root(x, K, tau, delta


impl_vol_bs <- simplify2array(map2(stock_prc_list, bs_res, function(x,y) vol_root(x, K, tau, delta, r, 

#impl volatility
ggplot() +
  geom_line(aes(x = stock_prc_list, y = impl_vol_hes, color = "Heston")) +
  geom_line(aes(x = stock_prc_list, y = impl_vol_bs, color = "Black-Scholes")) +
  theme_bw() + ylab("Implied Volatility") + xlab("Underlying Price") +
  ggtitle(TeX("Heston Implied Volatility $\\rho = -0.5$")) +
  scale_color_manual(name = "Pricer",
                     values = c("Heston" = "blue", "Black-Scholes" = "black"))

#rho = 0.5

heston_res_2 = sapply(stock_prc_list, function(x) heston_call_prc(x, K, tau, r, delta, vt, theta, rho =

impl_vol_hes_2 <- simplify2array(map2(stock_prc_list, heston_res_2, function(x,y) vol_root(x, K, tau, de


#impl volatility
ggplot() +
  geom_line(aes(x = stock_prc_list, y = impl_vol_hes_2, color = "Heston")) +
  geom_line(aes(x = stock_prc_list, y = impl_vol_bs, color = "Black-Scholes")) +
  theme_bw() + ylab("Implied Volatility") + xlab("Underlying Price") +
  ggtitle(TeX("Heston Implied Volatility $\\rho = 0.5$")) +
  scale_color_manual(name = "Pricer",
                     values = c("Heston" = "blue", "Black-Scholes" = "black"))
```