Project #5: Lognormal stock prices. Black-Scholes.

Samuel Kalisch

2024-04-29

```
library(ggplot2)
library(tidyverse)
library(scales)
library(plotly)
```

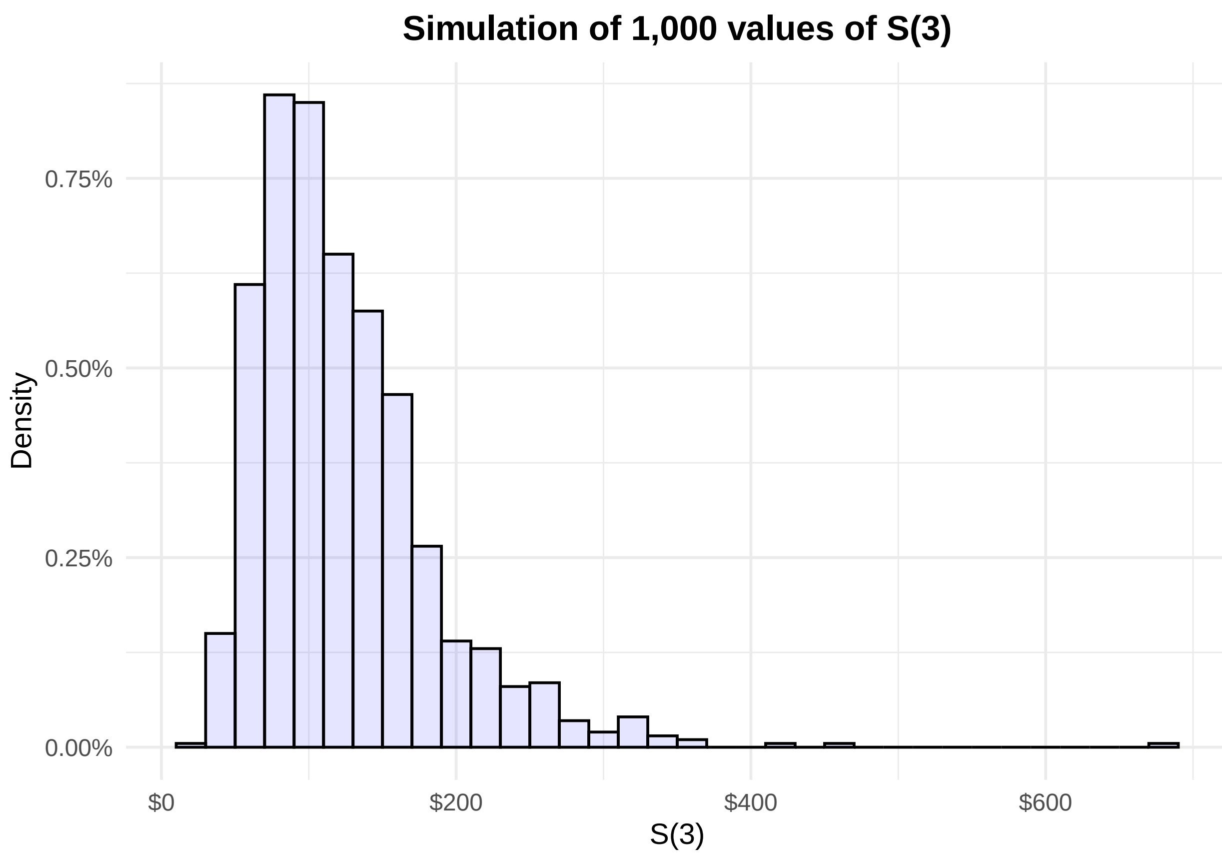
Problem #1 (25 points)

Assume the **Black-Scholes model**. The continuously compounded, risk-free interest rate is 0.06. Let the current stock price of a non-dividend-paying stock be $100. Let its volatility be 0.25.

```
r = 0.06
s0= 100
sigma=.25
```

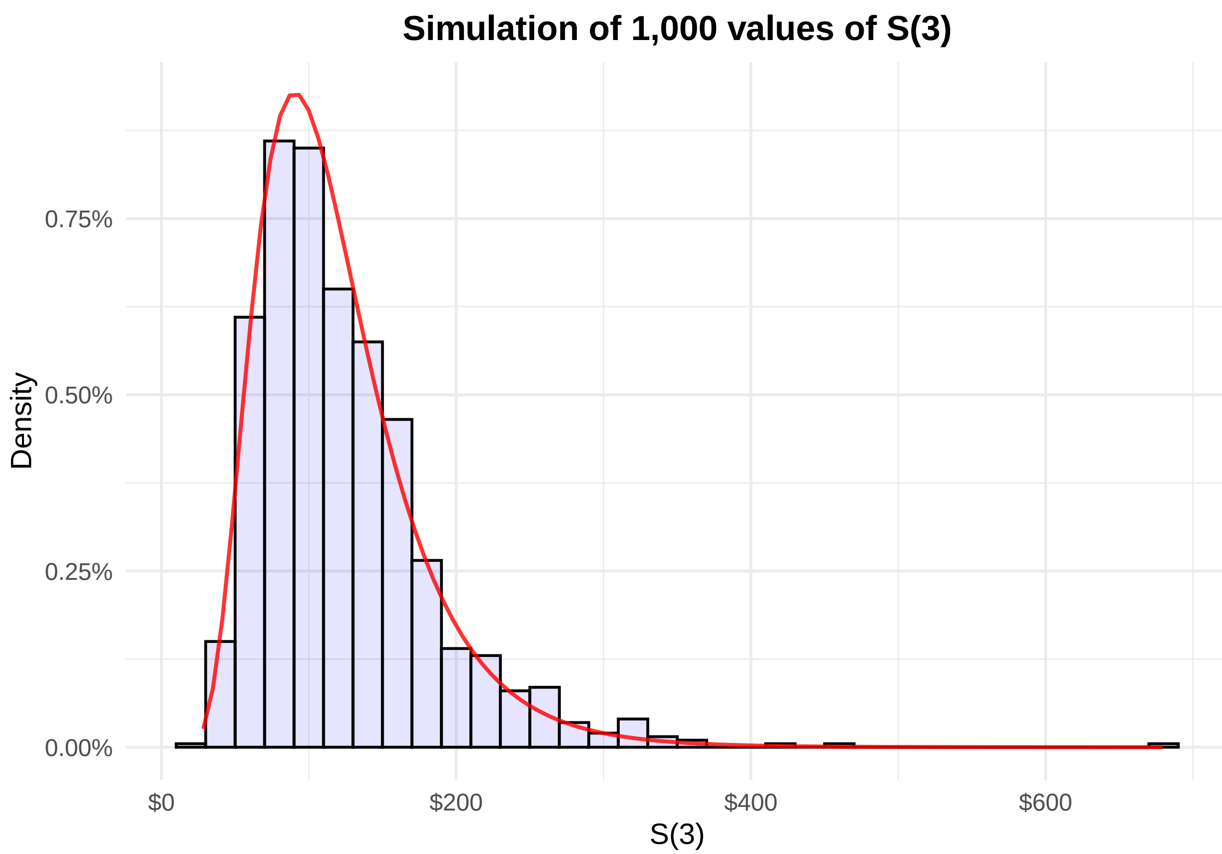
(5 points) Simulate 1000 values of the stock price at time $t = 3$. Draw a histogram of the obtained set of values.

```
nsims=10^3
T= 3

mu = r-sigma^2/2
nu = sigma*sqrt(T)
s.T= s0*rlnorm(n=nsims, meanlog = mu*T, sdlog=nu)

ggplot(data.frame(x = s.T), aes(x = x)) +
  geom_histogram(binwidth = 20, aes(y = after_stat(density)),
                 fill = "blue", color = "black", alpha = 0.1)+
  labs(title = paste("Simulation of 1,000 values of S(3)"),
       x = "S(3)",
       y = "Density") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5, face = "bold"))+
  scale_x_continuous(labels = dollar_format(prefix = "$")) +
  scale_y_continuous(labels = percent_format(accuracy = .01))
```



(6 points) Superimpose the graph of the density of a lognormal distribution of $S(3)$ with the appropriate parameters onto the histogram obtained above. The command `dlnorm` should come in handy.

```
ggplot(data.frame(x = s.T), aes(x = x)) +
  geom_histogram(binwidth = 20, aes(y = after_stat(density)),
                 fill = "blue", color = "black", alpha = 0.1)+
   stat_function(fun = dlnorm, args = list(mean = log(s0)+mu*T, sd = nu), color = "red",
                linewidth = .7, alpha = .8) +
  labs(title = paste("Simulation of 1,000 values of S(3)"),
        x = "S(3)",
        y = "Density") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5, face = "bold"))+
  scale_x_continuous(labels = dollar_format(prefix = "$")) +
  scale_y_continuous(labels = percent_format(accuracy = .01))
```



(2 points) Find the median of the set of simulated values you obtained above.

```
median(s.T)%>%dollar()
## [1] "$110.96"
```

(2 points) Find the mean of the set of simulated values you obtained above.

```
mean(s.T)%>%dollar()
## [1] "$124.11"
```

(2 points) What is the proportion of stock prices you created in this way that above \$105?

```
K = 105
mean(s.T>K)%>%percent(.01)
## [1] "54.00%"
```

(5 points) What is the theoretical risk-neutral probability that the final asset price exceeds \$105?

```
d2= (log(s0/K)+(r-sigma^2/2)*T)/sigma/sqrt(T)
pnorm(d2)%>%percent(.01)
## [1] "53.45%"
```

(3 points) Comment on the comparison between the proportion and the risk-neutral probability you obtained above. Which theorem from probability is useful here?

Law of Large Numbers (LLN)

The Law of Large Numbers (LLN) is fundamental in understanding the accuracy of the Monte Carlo method. It states that as the number of trials or samples (n) increases, the sample average converges to the expected value. Mathematically, for a sequence of independent and identically distributed random variables $X_1, X_2, \ldots, X_n$ with mean μ and finite variance, the LLN is expressed as:

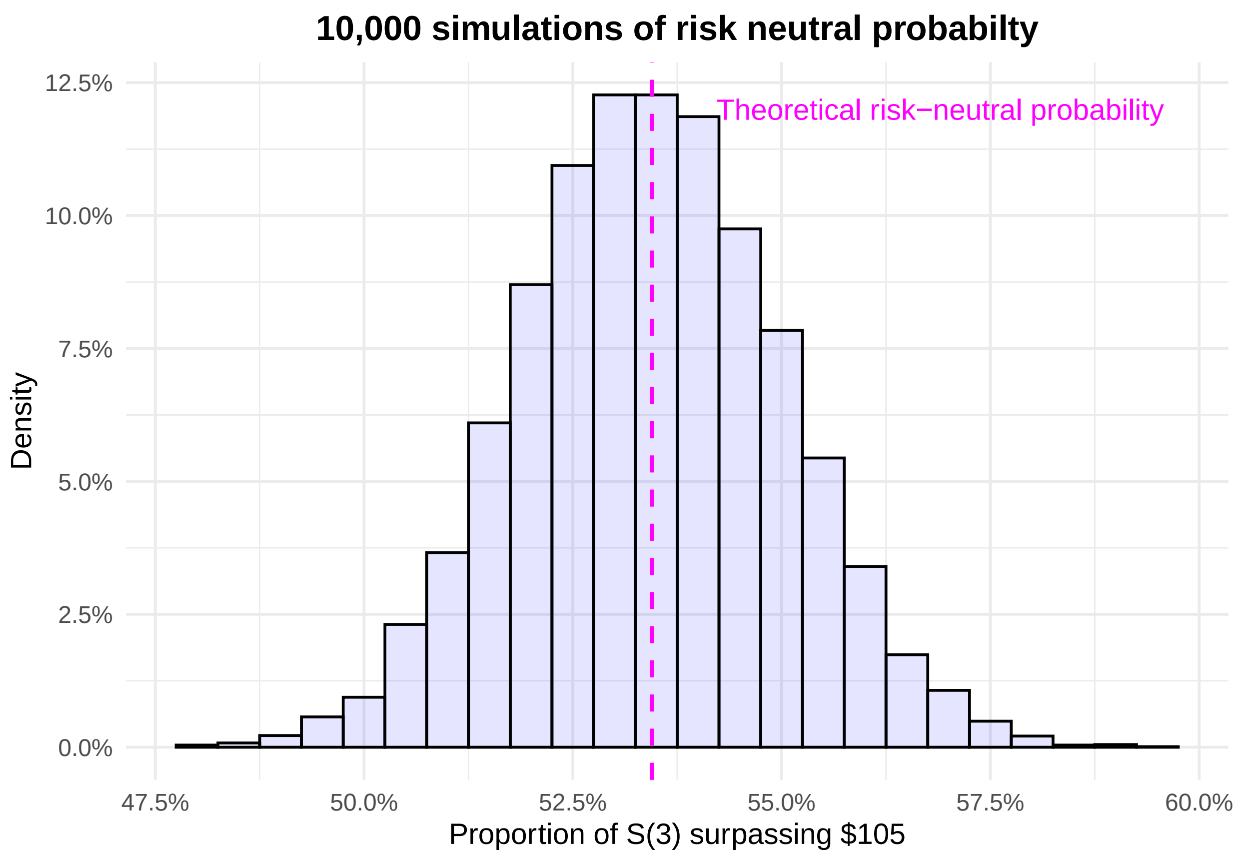
- $$\lim_{n \to \infty} \frac{1}{n} \sum_{i=1}^{n} X_i = \mu$$

By the properties of variance, $Var(\bar{X}_n) = \frac{\sigma^2}{n}$. As n approaches infinity, $Var(\bar{X}_n)$ approaches zero. This implies that $\bar{X}_n$ becomes increasingly concentrated around its mean μ as n increases, leading to the convergence of $\bar{X}_n$ to μ as n tends to infinity, as stated by the LLN.

Variance of 1,000 Simulations of S(3):

```r
n.runs= 10^4
runs = replicate(n.runs, mean(s0 * rlnorm(n = nsims, meanlog = mu * T, sdlog = nu) > K))

sd(runs)%>%percent(.01)
## [1] "1.56%"
```

```r
ggplot(data.frame(x = runs), aes(x = x)) +
  geom_histogram(binwidth = .005, aes(y = after_stat(count)/n.runs),
                 fill = "blue", color = "black", alpha = 0.1) +
  geom_vline(xintercept = pnorm(d2), color = "magenta",
             linetype = "dashed", linewidth = .75) +
  annotate("text", x =.569, y = .12, label = "Theoretical risk-neutral probability",
           color = "magenta") +
  labs(title = paste("10,000 simulations of risk neutral probabilty"),
       x = "Proportion of S(3) surpassing $105",
       y = "Density") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5, face = "bold"))+
  scale_x_continuous(labels = percent_format(accuracy = .1)) +
  scale_y_continuous(labels = percent_format(accuracy = .1))
```



Problem #2 (25 points)

Let the continuously compounded, risk-free interest rate be 0.05.

Consider a stock whose current price is \$120 and whose volatility is 0.20. We will be pricing a three-month, \$115-strike call option.

```
r = .05
s0 = 120
sigma = .2
K=115
T=3/12
```

Part a: Analytic Black-Scholes (5 points)

Price the option above using the Black-Scholes pricing formula.

```
blackScholes <- function(S, K, r, sigma, T, type = "c") {
  d1 <- (log(S/K) + (r + sigma^2/2) * T) / (sigma * sqrt(T))
  d2 <- d1 - sigma * sqrt(T)
  if (type == "c") {return(S * pnorm(d1) - K * exp(-r * T) * pnorm(d2))}
  else if (type == "p") {return( K * exp(-r * T) * pnorm(-d2) - S * pnorm(-d1))
  }
}
```

```r
blackScholes(s0, K, r, sigma, T)%>%dollar()
## [1] "$8.56"
```

Part b: Black-Scholes Monte Carlo with Z (10 points)

Provide the *Monte Carlo* estimate of the price using the simulated draws from the standard normal distribution with 10000 simulations.

```r
nsims = 10^4
z = rnorm(nsims)
mu = r-sigma^2/2
nu = sigma*sqrt(T)

s.T = s0* exp(mu*T+nu*z)
v.T = pmax(s.T-K,0)

mean(exp(-r*T)*v.T)%>%dollar()
## [1] "$8.26"
```

Part c: Black-Scholes Monte Carlo with 'rlnorm" (10 points)

Provide the *Monte Carlo* estimate of the price using the simulated draws from the lognormal distribution with 10000 simulations.

```r
nsims = 10^4
mu = r-sigma^2/2
nu = sigma*sqrt(T)
logn = rlnorm(nsims,mu*T,nu)

s.T = s0*logn
v.T = pmax(s.T-K,0)

mean(exp(-r*T)*v.T)%>%dollar()
## [1] "$8.66"
```

Problem #3 (25 points)

Let the continuously compounded, risk-free interest rate be 0.03.

Consider a stock whose current price is \$90 and whose volatility is 0.30. We will be pricing a half-year, \$95-strike put option.

```r
r = .03
s0 = 90
sigma = .3
K=95
T=1/2
```

Part a: Analytic Black-Scholes (5 points)

Price the option above using the Black-Scholes pricing formula.

```
blackScholes(s0, K, r, sigma, T,"p")%>%dollar()
## [1] "$9.68"
```

Part b: Black-Scholes Monte Carlo with Z (10 points)

Provide the *Monte Carlo* estimate of the price using the simulated draws from the standard normal distribution with 10000 simulations.

```
nsims = 10^4
z = rnorm(nsims)
mu = r-sigma^2/2
nu = sigma*sqrt(T)

s.T = s0* exp(mu*T+nu*z)
v.T = pmax(K-s.T,0)

mean(exp(-r*T)*v.T)%>%dollar()
## [1] "$9.64"
```

Part c: Black-Scholes Monte Carlo with 'rlnorm" (10 points)

Provide the *Monte Carlo* estimate of the price using the simulated draws from the lognormal distribution with 10000 simulations.

```
nsims = 10^4
mu = r-sigma^2/2
nu = sigma*sqrt(T)
logn = rlnorm(nsims,mu*T,nu)

s.T = s0*logn
v.T = pmax(K-s.T,0)

mean(exp(-r*T)*v.T)%>%dollar()
## [1] "$9.90"
```

Problem #4 (25 points)

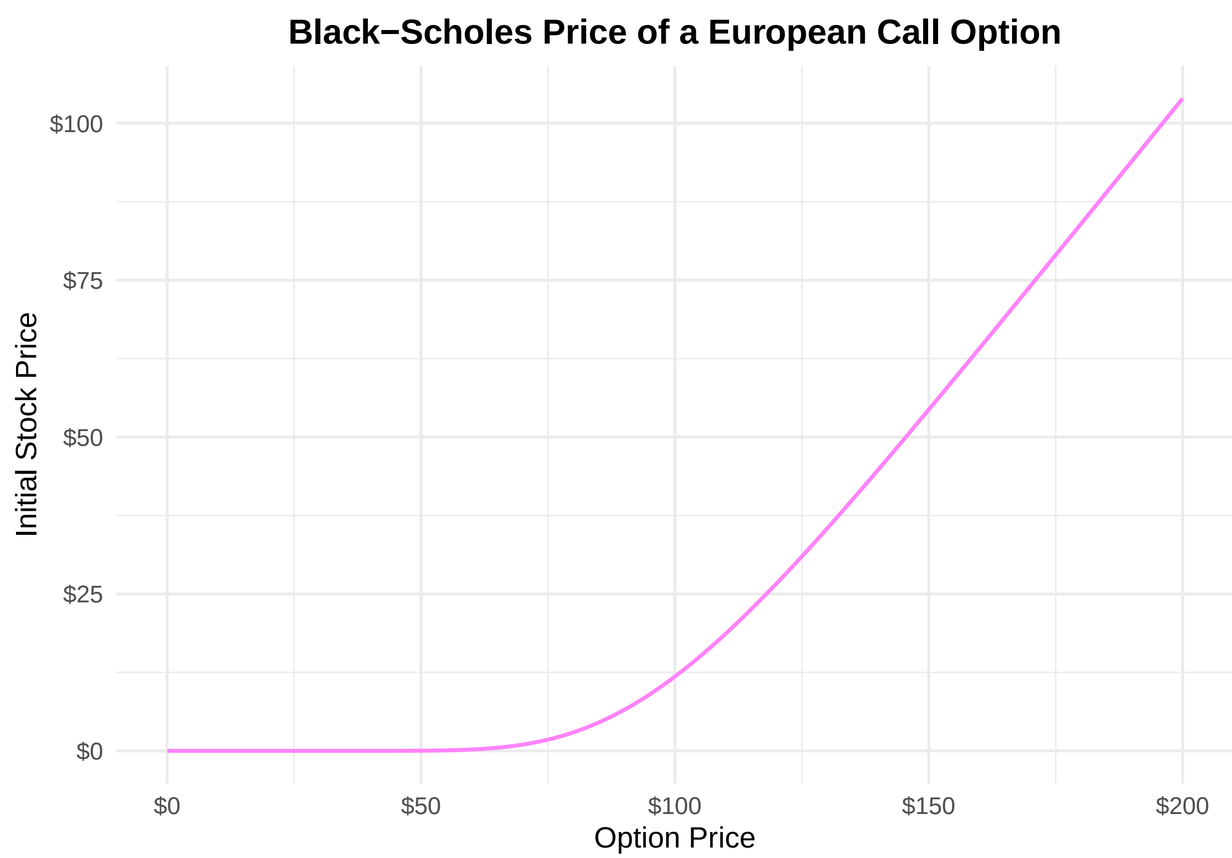
Let the continuously compounded, risk-free interest rate be 0.04.

Consider a stock whose volatility is 0.25.

```
r = .04
sigma = .25
```

(5 points) First, we will be focusing on a one-year, $100-strike European call option. Define a function which calculates the analytic Black-Scholes price of this call as a function of the initial asset price denoted by s. Then, plot the graph of that function. Let the domain of your plot be $[0, 200]$.

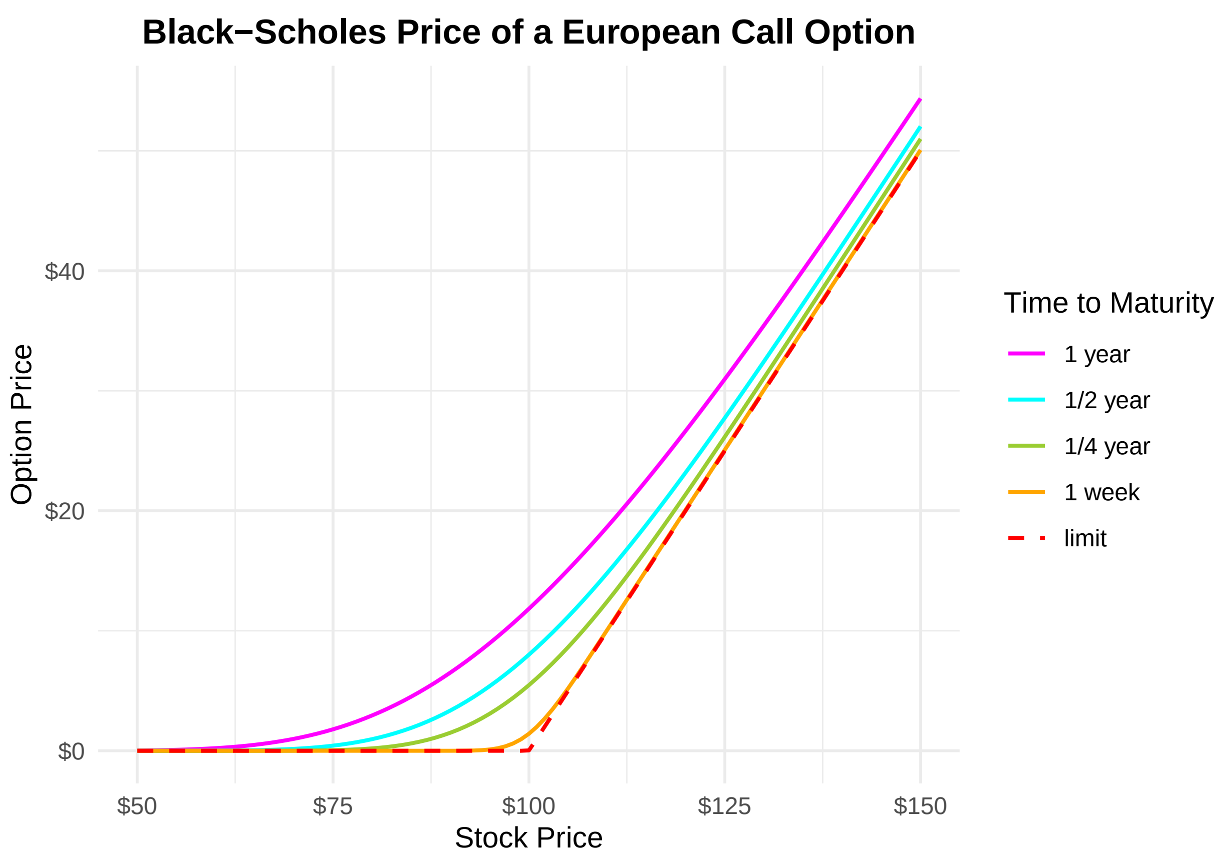
```r
#funciton is defined in probelm 2
K=100

s.0 =seq(0, 200, by = 1)
v.0=blackScholes(s.0, K, r, sigma, T=1)

#plotting
ggplot(data = data.frame(s.0, v.0), aes(x = s.0, y = v.0)) +
geom_line(color = "orchid1", linewidth = 0.7) +
labs(title = "Black-Scholes Price of a European Call Option",
x = "Option Price",
y = "Initial Stock Price") +
theme_minimal()+
  theme(plot.title = element_text(hjust = 0.5, face = "bold"))+
  scale_x_continuous(labels = scales::dollar_format(big.mark = ","))+
  scale_y_continuous(labels = scales::dollar_format(big.mark = ","))
```



(10 points) Now, plot the graph of the above Black-Scholes price (as a function of the underlying stock price) one year, half a year, quarter year and a week prior to exercise date. Add the plot of the payoff function. Let the domain of your plot be $[50, 150]$. Make sure that all the plots are in different colors. What do you notice as the exercise date approaches? What can you say about the call delta?

```r
s <- seq(50, 150, by = 1)
v <- sapply(c(1, 1/2, 1/4, 7/365,.00001), function(T) blackScholes(s, K, r, sigma, T))
colors <- c("magenta", "cyan", "yellowgreen", "orange","red1")
```

```
time_to_maturity <- c("1 year", "1/2 year", "1/4 year", "1 week","limit")

# Wrangle data into a data frame
df <- data.frame(s = rep(s, 5),
  v = c(v[,1], v[,2], v[,3], v[,4], v[,5]),
  Time_to_Maturity = factor(rep(time_to_maturity, each = length(s)),
                            levels = time_to_maturity))

# Plotting using ggplot
ggplot(df, aes(x = s, y = v, color = Time_to_Maturity, group = Time_to_Maturity)) +
  geom_line(data = df[df$Time_to_Maturity != "limit", ], linewidth = 0.7) +
  geom_line(data = df[df$Time_to_Maturity == "limit", ], linewidth = 0.7, linetype = "dashed") +
  scale_color_manual(values = colors) +
  labs(title = "Black-Scholes Price of a European Call Option",
       x = "Stock Price",
       y = "Option Price",
       color = "Time to Maturity") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5, face = "bold")) +
  scale_x_continuous(labels = scales::dollar_format(big.mark = ",")) +
  scale_y_continuous(labels = scales::dollar_format(big.mark = ","))
```



Observations:

1. The option prices for longer times to maturity are higher. This is because the more time there is until

the option expires, the greater the chance the stock price will move in a favorable direction, and thus, the option carries a higher time value.

2. As the exercise date approaches, the option prices for different stock prices converge closer to the intrinsic value of the option. For example, the curve for 1 week to maturity is much steeper and closer to the 'hockey stick' shape of the intrinsic value of a call option.

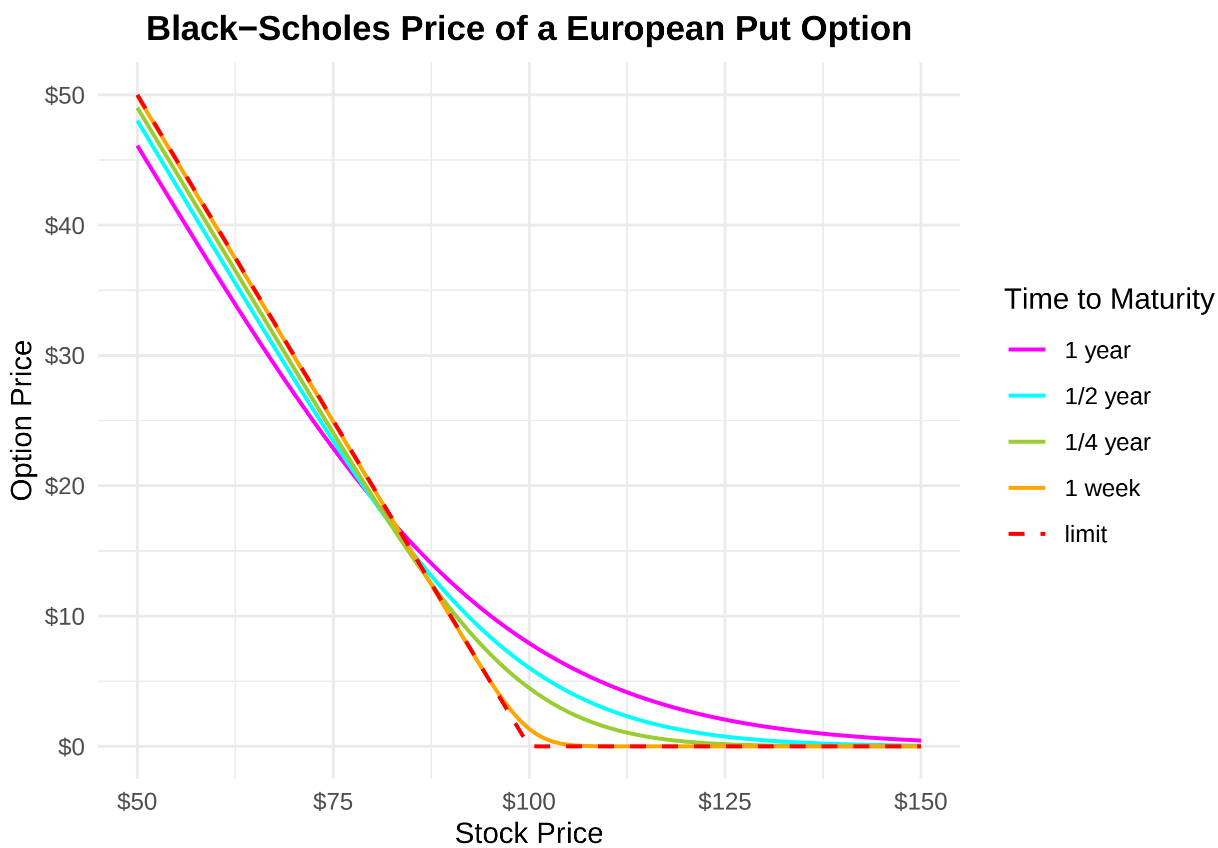
Delta:

1. For the 1-week curve, the delta is closer to 0 for stock prices well below the strike price, meaning the option's price is less sensitive to stock price changes because it is likely to expire worthless. However, as the stock price increases and moves in-the-money, the delta increases and approaches 1, indicating that the option's price will move nearly dollar-for-dollar with the stock price since it is more likely to expire in-the-money.

2. For options with longer times to maturity, the delta changes more gradually. This is because there's more time for the stock price to move, so the immediate sensitivity to stock price changes is lower. The delta for these options will be between 0 and 1 for a wider range of stock prices, reflecting the uncertain outcome of the option at expiration

(10 points) Repeat the previous problem for an otherwise identical put option.

```r
s <- seq(50, 150, by = 1)
v <- sapply(c(1, 1/2, 1/4, 7/365,.00001), function(T) blackScholes(s, K, r, sigma, T,"p"))
colors <- c("magenta", "cyan", "yellowgreen", "orange","red1")
time_to_maturity <- c("1 year", "1/2 year", "1/4 year", "1 week","limit")

# Wrangle data into a data frame
df <- data.frame(s = rep(s, 5),
  v = c(v[,1], v[,2], v[,3], v[,4], v[,5]),
  Time_to_Maturity = factor(rep(time_to_maturity, each = length(s)),
                            levels = time_to_maturity))

# Plotting using ggplot
ggplot(df, aes(x = s, y = v, color = Time_to_Maturity, group = Time_to_Maturity)) +
  geom_line(data = df[df$Time_to_Maturity != "limit", ], linewidth = 0.7) +
  geom_line(data = df[df$Time_to_Maturity == "limit", ], linewidth = 0.7, linetype = "dashed") +
  scale_color_manual(values = colors) +
  labs(title = "Black-Scholes Price of a European Put Option",
       x = "Stock Price",
       y = "Option Price",
       color = "Time to Maturity") +
  theme_minimal() +
  theme(plot.title = element_text(hjust = 0.5, face = "bold")) +
  scale_x_continuous(labels = scales::dollar_format(big.mark = ",")) +
  scale_y_continuous(labels = scales::dollar_format(big.mark = ","))
```



Observations:

1. The option prices for longer times to maturity are higher until they become far out of the money. Then they have a positive theta

- "Instances of negative time value and, consequently, positive theta are relaively rare. At a minimum, the option must be subject to stock-type settlement, it must be deeply in the money, and it must also be European with no possibility of early exercise. If the option were American, everyone would exercise it today in order to earn interest on the intrinsic value." *Option Volatility and Pricing Advanced Trading Strategies and Techniques, pg. 109*

2. As the exercise date approaches, the option prices for different stock prices converge closer to the intrinsic value of the option. For example, the curve for 1 week to maturity is much steeper and closer to the 'hockey stick' shape of the intrinsic value of a put option.

Delta:

1. For the 1-week curve, the delta is closer to 0 for stock prices well above the strike price, meaning the option's price is less sensitive to stock price changes because it is likely to expire worthless. However, as the stock price decreases and moves in-the-money, the delta increases and approaches -1, indicating that the option's price will move nearly dollar-for-dollar with the stock price since it is more likely to expire in-the-money.
2. For options with longer times to maturity, the delta changes more gradually. This is because there's more time for the stock price to move, so the immediate sensitivity to stock price changes is lower.

The delta for these options will be between 0 and -1 for a wider range of stock prices, reflecting the uncertain outcome of the option at expiration

3-d graphs [extra]:

```
# Parameters
S <- seq(50, 150, length.out = 100)
T <- seq(0.00001, 1, length.out = 100)

V <- outer(S, T, Vectorize(function(s, t) blackScholes(s, K, r, sigma, t)))

# Create a surface plot
fig <- plot_ly(x = ~T, y = ~S, z = ~V, type = "surface")
fig <- fig %>% layout(title = "<b>Black-Scholes Price of a Call Option",
                      scene = list(xaxis = list(title = "Time to Maturity"),
                                   yaxis = list(title = "Stock Price"),
                                   zaxis = list(title = "Option Price")))
```

```
# Parameters
S <- seq(50, 150, length.out = 100)
T <- seq(0.00001, 1, length.out = 100)

V <- outer(S, T, Vectorize(function(s, t) blackScholes(s, K, r, sigma, t, type="p")))

# Create a surface plot
fig2 <- plot_ly(x = ~rev(T), y = ~S, z = ~V, type = "surface")
fig2 <- fig2 %>% layout(title = "<b>Black-Scholes Price of a Put Option",
                      scene = list(xaxis = list(title = "Time to Maturity"),
                                   yaxis = list(title = "Stock Price"),
                                   zaxis = list(title = "Option Price")))
```

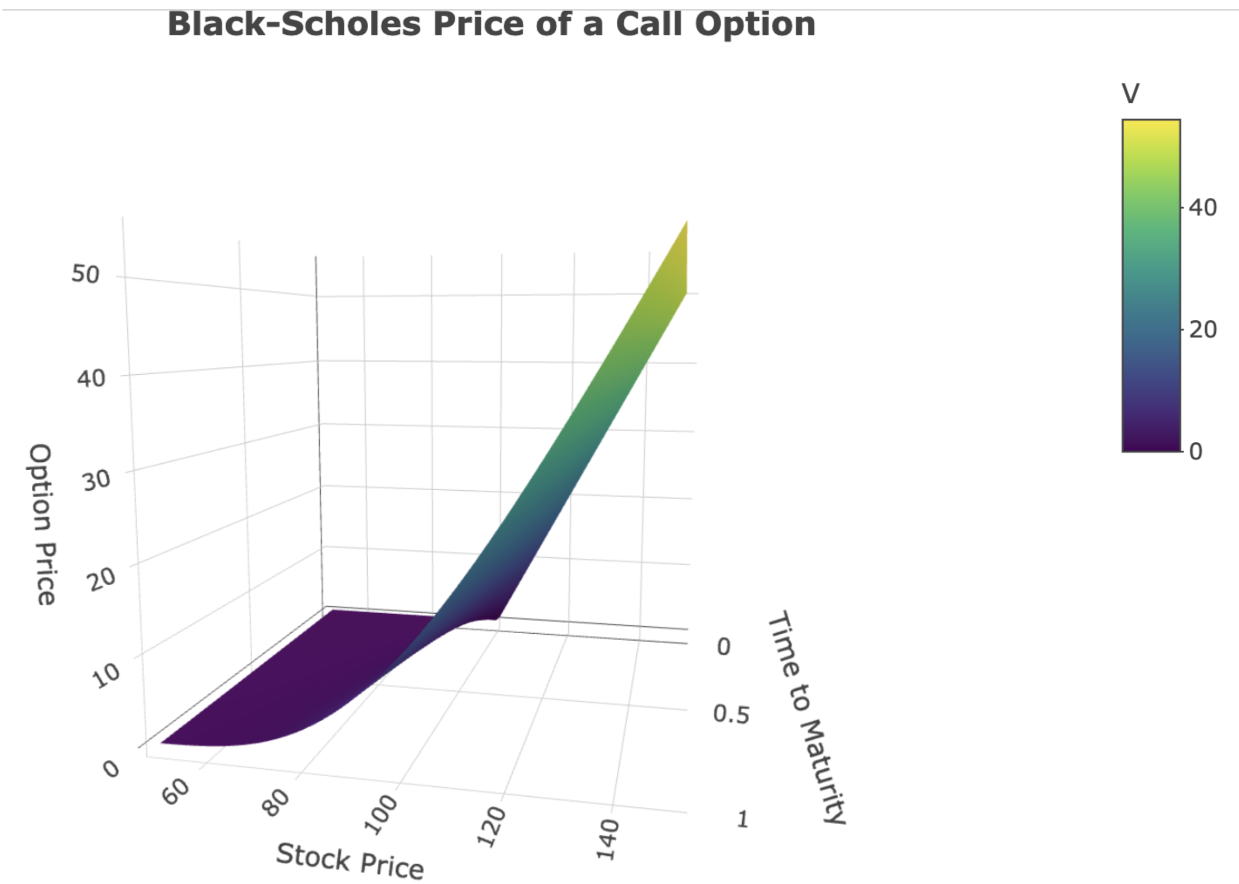


Figure 1: Black-Scholes Price of a Call Option

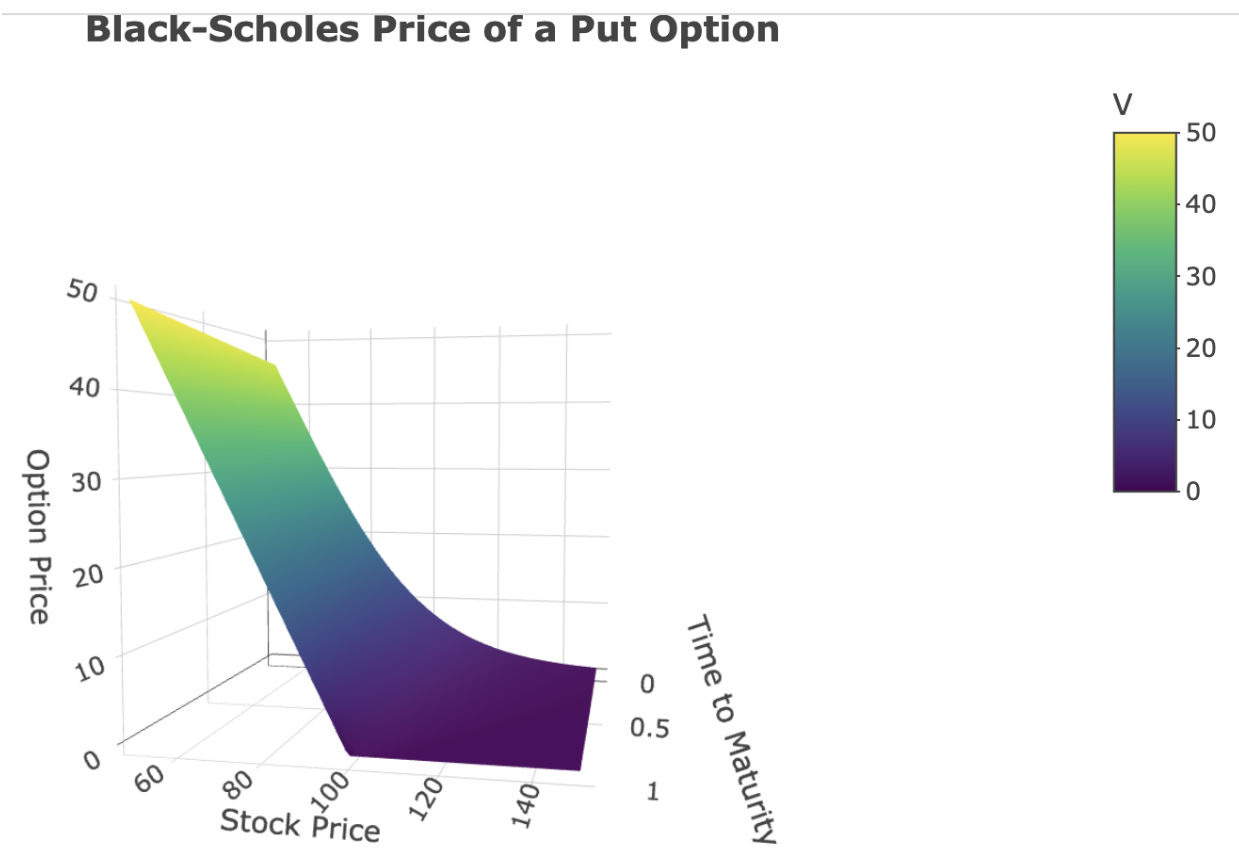


Figure 2: Black-Scholes Price of a Put Option