# Black-Scholes

Marco e Victor

```r
library(tidyverse)
library(gridExtra)
```

## European Options

### Total Value

Total.value = Intrinsic.value + Time.value

Intrinsic value: the maximum of zero and the payoff from the option if it were exercised immediately.

Call: max(S–K,0)

Put: max(K–S,0)

Time value: The excess of an option's value over its intrinsic value

### Moneyness

- At-the-money (**ATM**): S = K
- In-the-money (**ITM**): Option has intrinsic value > 0
- Out-of-the-money (**OOTM**): Option has instrinsic value = 0

```r
# Assuming a $50 strike price (k)
k <- 50
# We define a vector of potential terminal spot prices (st)
st <- 0:100

# PO = Payoff
# We calculate the potential call and put options payoffs
c_po <- pmax(st - k, 0)

p_po <- pmax(k-st,0)

po <- data.frame(st, c_po, p_po)
```

### Graph of each combination of option type

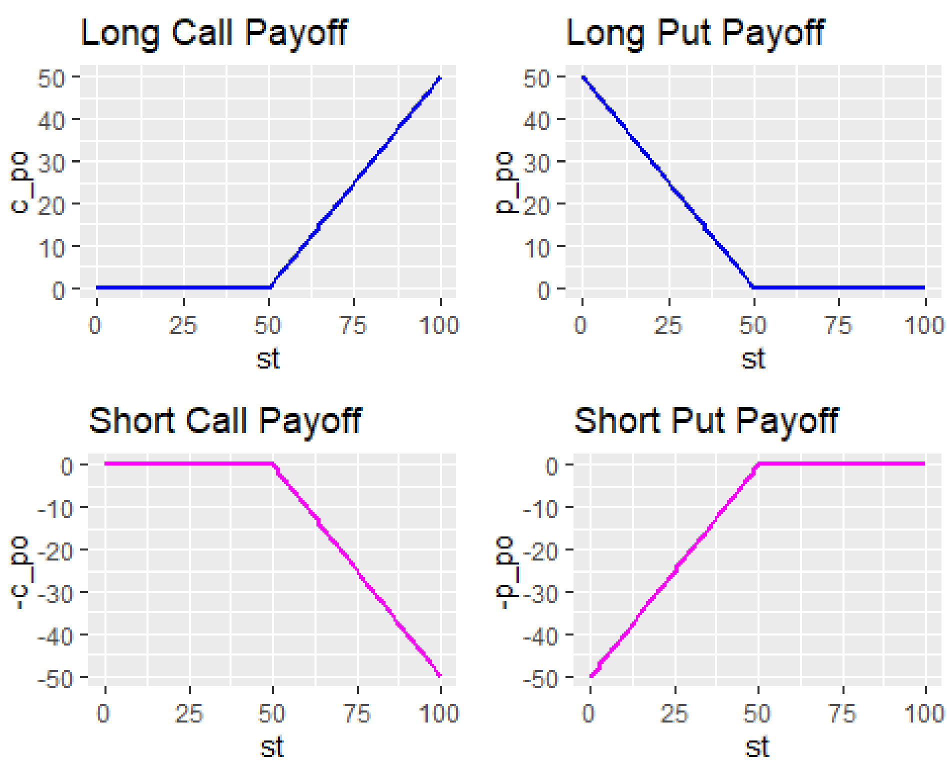
```r
lcpo <- ggplot(po, aes(x = st, y = c_po)) +
  geom_line(color = "blue", linewidth = 1) +
  ggtitle("Long Call Payoff")
```

```
lppo <- ggplot(po, aes(x = st, y = p_po)) +
  geom_line(color = "blue",linewidth = 1) +
  ggtitle("Long Put Payoff")

scpo <- ggplot(po, aes(x = st, y = -c_po)) +
  geom_line(color = "magenta", linewidth = 1) +
  ggtitle("Short Call Payoff")

sppo <- ggplot(po, aes(x = st, y = -p_po)) +
  geom_line(color = "magenta", linewidth = 1) +
  ggtitle("Short Put Payoff")

grid.arrange(lcpo, lppo, scpo, sppo, nrow = 2)
```




## Assume call and put costs

```
call = 10
put = 10
c_np <- c_po - call
p_np <- p_po - put
np <- data.frame(st, c_np, p_np)



lcnp <- ggplot(np, aes(x = st, y = c_po)) +
  geom_line(color = "blue", linewidth = 1) +
  geom_line(y = c_np, linetype = "dashed") +
```

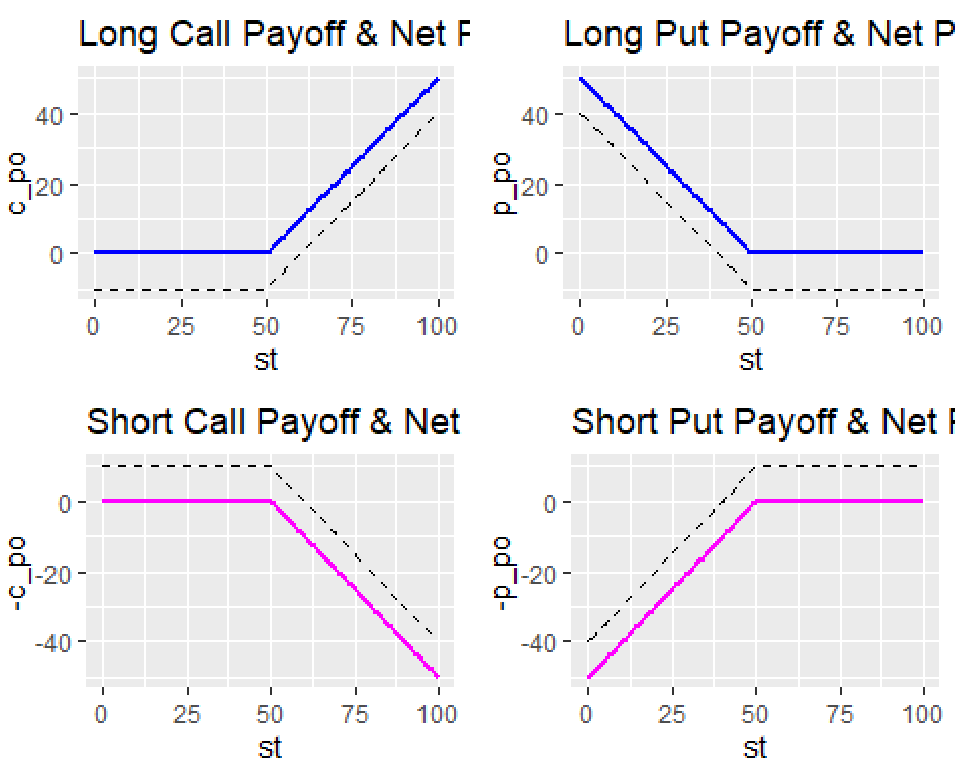
```
  ggtitle("Long Call Payoff & Net Profit") +
  ylim(-10, 50)

lpnp <- ggplot(np, aes(x = st, y = p_po)) +
  geom_line(color = "blue", linewidth = 1) +
  geom_line(y = p_np, linetype = "dashed") +
  ggtitle("Long Put Payoff & Net Profit") +
  ylim(-10, 50)

scnp <- ggplot(np, aes(x = st, y = -c_po)) +
  geom_line(color = "magenta", linewidth = 1) +
  geom_line(y = -c_np, linetype = "dashed") +
  ggtitle("Short Call Payoff & Net Profit") +
  ylim(-50, 10)

spnp <- ggplot(np, aes(x = st, y = -p_po)) +
  geom_line(color = "magenta", linewidth = 1)  +
  geom_line(y = -p_np, linetype = "dashed") +
  ggtitle("Short Put Payoff & Net Profit") +
  ylim(-50, 10)

grid.arrange(lcnp, lpnp, scnp, spnp, nrow = 2)
```



## Monte Carlo Simulation

- Involves generating multiple (10^06 or more) random values corresponding to the independent variable, based on an assumed probability distribution.
- For stock options valuation we want to model the price of the underlying asset, so the random values are plugged in the corresponding valuation function.
- We then average the potential values of all scenarios, and bring that expected value to the present.

### Theoretical Distribution of Stock Prices

Given by:

$ST=S0e[(\mu-\sigma^2/2)T+\sigma\epsilon\sqrt{T}]$

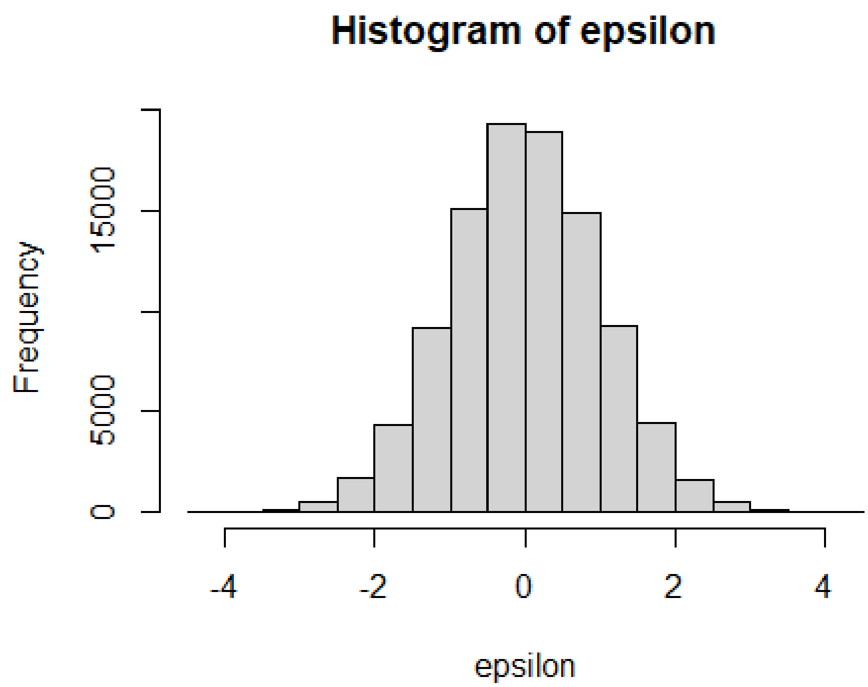
$\mu$ is the risk free rate (the "drift") and $\epsilon$ is a random number with Standard Normal Distribution ~N(0,1) (the "stochastic" component, Wiener process, Brownian motion, etc). Given that we are assuming that **Returns follow a Normal Distribution**, then **Prices follow a Lognormal Distribution** (they are always positive).

### Step-by-Step Computation

**Input data**

- type: european call or put
- s: underlying asset's spot price
- k: option's strike price
- t: option's time to maturity in years (e.g. 3 months = 3/12 years)
- v: underlying asset's returns annualized volatility
- r: annualized risk-free rate
- m: number of simulations

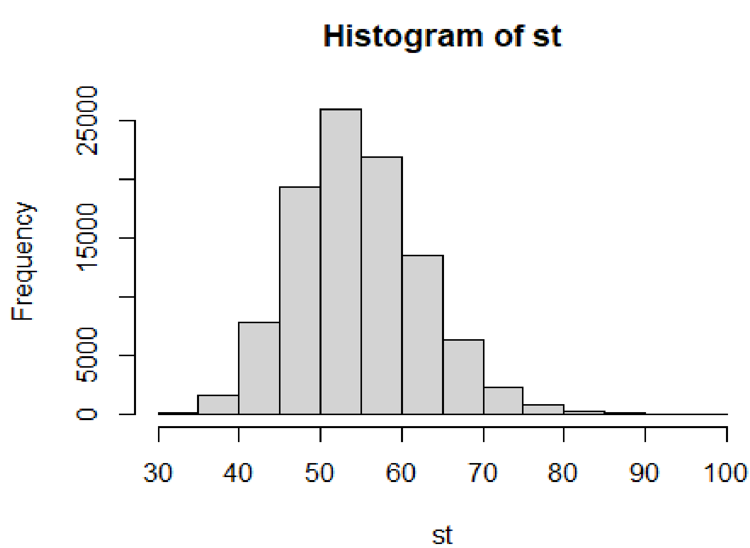
```
# Input data
m = 100000
type = "c"
s = 52
k = 50
t = 0.50
v = 0.20
r = 0.10
# Generation of stochastic component
epsilon <- rnorm(m, 0, 1)
hist(epsilon)
```



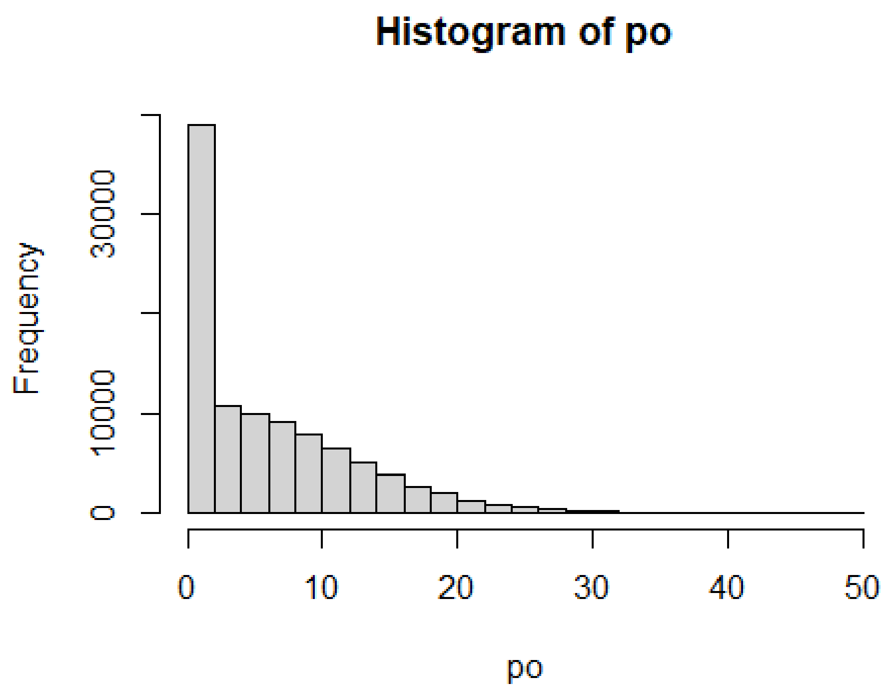

```
# Price simulation
st <- s*exp((r - (v^2)/2)*t + v * epsilon*sqrt(t))
print(paste("Min St:", round(min(st), 2),
            "/ Avg St:", round(mean(st), 2),
            "/ Max St:", round(max(st), 2)))

[1] "Min St: 30.07 / Avg St: 54.64 / Max St: 99.69"

hist(st)
```



```
# Payoff estimations for call or put
if(type == "c") po <- pmax(st - k, 0)
if(type == "p") po <- pmax(k-st, 0)
hist(po)
```




```
# Risk-neutral stock price at T (Future)
round(s*exp(r*t), 2)

[1] 54.67

# Scenarios
scenarios <- data.frame(epsilon, st, po)
scenarios[sample(nrow(scenarios), 5),]

          epsilon       st        po
2247  -0.36993084 51.36349  1.363489
83432  1.03437288 62.64765 12.647648
99083 -0.05161002 53.72857  3.728574
61052 -0.64349293 49.41431  0.000000
98308 -0.27675466 52.04479  2.044790

# Option price
op <- mean(po) * exp(-r*t) # bring to present value the mean of all cases
simulated
op

[1] 5.541914
```

## Custom Function

We can define a custom function so that we can apply the method to a variety of options quickly.

```
# Function definition
mc <- function(m = 100000, type = "c", s = 52, k = 50,
               t = 0.50, v = 0.20, r = 0.10) {

  epsilon <- rnorm(m, 0, 1)
  st <- s*exp((r - (v^2)/2)*t + v*epsilon*sqrt(t))

  if(type == "c") po <- pmax(st - k, 0)
  if(type == "p") po <- pmax(k - st, 0)

  op <- mean(po) * exp(-r*t)
  return(op)
}
mc()

[1] 5.55611

mc(type = 'p')

[1] 1.128952
```

# Black-Scholes

Given that **European Options** can only be exercised at maturity, the only thing that matters is the prices distribution at time T.

This option is NOT path-dependant, that is, it only matters the final ST, not how it got there.

Hence, we can "skip" the simulation part by using Black-Scholes equations.

## Equations

**Call price:**

c = S0*N(d1)* – *K*exp(−rT)*N(d2)

**Put price:**

p = K*exp(−rT)*N(−d2) – S0*N(−d1)

Where N(x) is the Standard Normal Cumulative Distribution Function (cdf) for x.

d1 = [ln(S0/K) + (r+σ^2/2)*T] / (σ√T)

$d2 = d1 - \sigma \sqrt{T}$

## Step-by-Step Computation

### Input Data

- type: call or put
- s: underlying asset's spot price
- k: option's strike price
- t: option's time to maturity in years (e.g. 3 months = 3/12 years)
- v: underlying asset's returns annualized volatility
- r: annualized risk-free rate

```
bs <- function(type = "c", s = 52, k = 50,
               t = 0.5, v = 0.20, r = 0.10) {

  op <- NA
  d1 <- (log(s/k)+(r+(v^2)/2)*t)/(v*sqrt(t))
  d2 <- d1 - v*sqrt(t)
  if(type == "c")   {
    op <- s*pnorm(d1) - k*exp(-r*t)*pnorm(d2)
  }
  if(type == "p") {
    op <- k*exp(-r*t)*pnorm(-d2) - s*pnorm(-d1)
  }
  return(op)
}
```

# Sensitivity Analysis

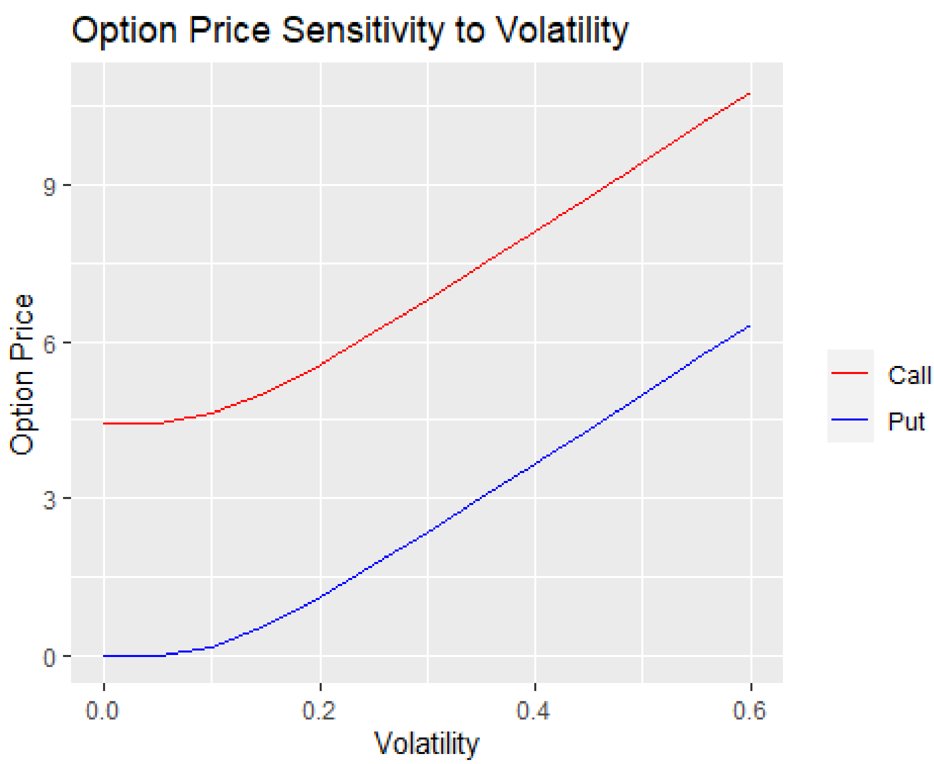
## Volatility

```
# Construction of Volatility Vector
vv <- seq(from=0.00, to=0.60, by=0.05)
vv

 [1] 0.00 0.05 0.10 0.15 0.20 0.25 0.30 0.35 0.40 0.45 0.50 0.55 0.60

# Call and Put Prices Vector: Using a For Loop
cv <- 0
pv <- 0
vv <- seq(from=0.00, to=0.60, by=0.05)
for (i in 1:length(vv)) {
  cv[i] <- bs(type= "c", v = vv[i])
  pv[i] <- bs(type = "p",v = vv[i])
}
```

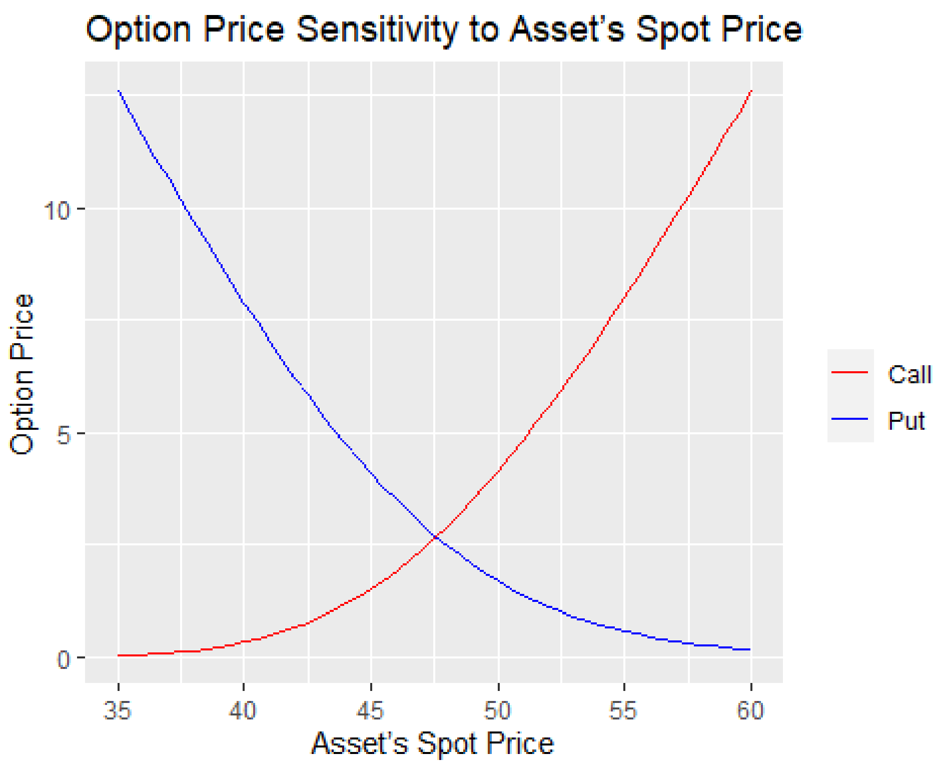
```r
# Sensitivity Volatility Plot
ggplot(data.frame(vv, cv, pv), aes(x = vv)) +
  geom_line(aes(y = cv, color = "Call")) +
  geom_line(aes(y = pv, color = "Put")) +
  scale_color_manual(values=c("red", "blue")) +
  labs(title = "Option Price Sensitivity to Volatility",
       color = "", y = "Option Price", x = "Volatility")
```




## Asset´s Spot Price

```r
# Construction of Asset’s Spot Price vector
ss <- seq(from=35, to=60, by=0.5)
cv <- pv <- 0
cv <- bs(type = "c",s=ss)
pv <- bs(type = "p",s=ss)

# Sensitivity Asset’s Spot Price Plot
ggplot(data.frame(ss, cv, pv), aes(x = ss)) +
  geom_line(aes(y = cv, color = "Call")) +
  geom_line(aes(y = pv, color = "Put")) +
  scale_color_manual(values=c("red", "blue")) +
  labs(title = "Option Price Sensitivity to Asset’s Spot Price",
       color = "", y = "Option Price", x = "Asset’s Spot Price")
```



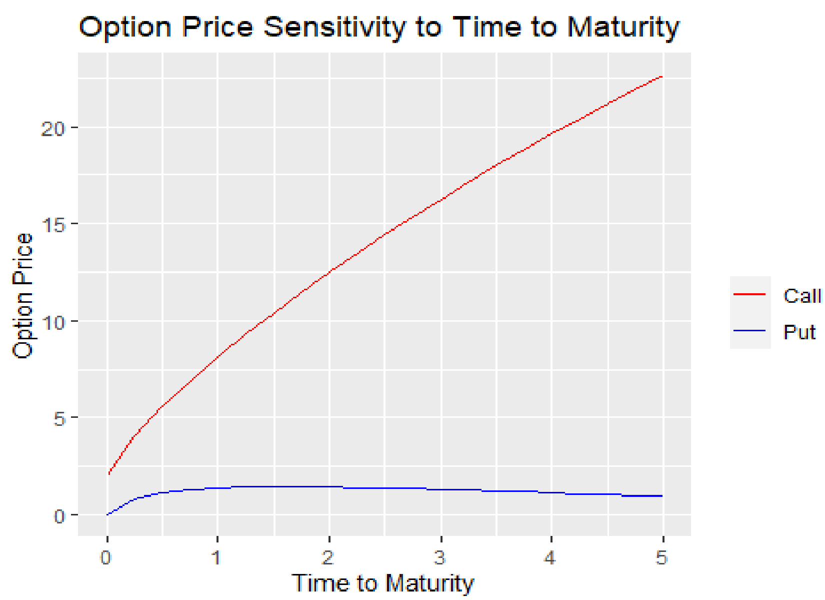

### Strike Price

```
# Construction of Strike Price vector
kk <- seq(from=35, to=70, by=0.5)
cv <- pv <- 0
cv <- bs(type = "c",k=kk)
pv <- bs(type = "p",k=kk)

# Sensitivity Asset’s Spot Price Plot
ggplot(data.frame(kk, cv, pv), aes(x = kk)) +
  geom_line(aes(y = cv, color = "Call")) +
  geom_line(aes(y = pv, color = "Put")) +
  scale_color_manual(values=c("red", "blue")) +
  labs(title = "Option Price Sensitivity to Strike Price",
       color = "", y = "Option Price", x = "Strike Price")
```


## Time to Maturity

```r
# Construction of Time to Maturity vector
tm <- seq(from = 0, to = 5, by = 0.25)
cv <- bs(type= 'c' ,t = tm)
pv <- bs(type = 'p', t = tm)
# Sensitivity Plot
ggplot(data.frame(tm, cv, pv), aes(x = tm)) +
  geom_line(aes(y = cv, color = "Call")) +
  geom_line(aes(y = pv, color = "Put")) +
  scale_color_manual(values=c("red", "blue")) +
  labs(title = "Option Price Sensitivity to Time to Maturity",
       color = "", y = "Option Price", x = "Time to Maturity")
```




## Risk Free Rate

```
# Construction of Risk Free Rate vector

rf <- seq(from = 0, to = 0.5, by = 0.01)
cv <- bs(type= 'c' ,r = rf)
pv <- bs(type = 'p', r = rf)
# Sensitivity Plot
ggplot(data.frame(rf, cv, pv), aes(x = rf)) +
  geom_line(aes(y = cv, color = "Call")) +
  geom_line(aes(y = pv, color = "Put")) +
  scale_color_manual(values=c("red", "blue")) +
  labs(title = "Option Price Sensitivity to Risk Free Rate",
       color = "", y = "Option Price", x = "Risk Free Rate")
```