HW 1
MFE 431: Financial Risk Management
Professor Valentin Haddad
Student: Xiahao Wang

Qn 1 VaR for an exponential distribution

1.

General formula:

$$prob(w < W_0 - VaR) = 1 - c$$

Hence:

$$W_0 - VaR = F^{-1}(1 - c)$$

2.

Apply with W_0 =200 and c = 99.9%

```
w0 <- 200
lambda <- 1/200
var <- w0 - (-log(1 - 0.001)/lambda)
cat("VaR is: ", var)
```

```
## VaR is:  199.7999
```

3.

```
var <- (-log(0.001)/lambda) - w0
cat("VaR is: ", var)
```

```
## VaR is:  1181.551
```

Unlike normal distribution, exponential distribution is not symmetrical, hence hence the VaR for longing and shorting are different.

4.

Expected Shortfall: let: $\lambda = 1/W_0$

$$ES = W_0 - \frac{\int_0^{W_0 - VaR} W\lambda e^{-\lambda W} dW}{\int_0^{W_0 - VaR} \lambda e^{-\lambda W} dW}$$

for long:

$$ES = W_0 - \frac{[-e^{-\frac{W}{200}}(W + 200)]_0^{0.2}}{0.001}$$

```
es_long <- w0 - (-exp(-0.2/200) * (200.2) - (-exp(0/200)*(200)))/0.001
cat("Expected Short Fall long: ", es_long)
```

```
## Expected Short Fall long:  199.9001
```

for short:

$$ES = W_0 - \frac{[-e^{-\frac{W}{200}}(W + 200)]_{1381.551}^{+\infty}}{0.001}$$

```
es_short <- abs(w0 - (0 + exp(-1381.551/200) * (1381.551 + 200)) / 0.001)
cat("Expected Short Fall short: ", es_short)
```

```
## Expected Short Fall short:  1381.551
```

Qn 2 VaR for mixtures

$W_0 = 1$ Unit is in billion for this question

1.

```
w0 <- 1
sigma1 <- 0.12
sigma2 <- 0.2
mu <- 0.08
pie <- 0.7
```

Partner view:

```
VaR <- qnorm(0.01, mu * w0, sigma1*w0)
VaR <- abs(VaR - mu)
cat("Parnter view VaR: ", VaR)
```

```
## Parnter view VaR:  0.2791617
```

My view:

```
VaR<- qnorm(0.01, mu * w0, sigma2 * w0)
VaR <- abs(VaR - mu)
cat("My view VaR: ", VaR)
```

```
## My view VaR:  0.4652696
```

Combining the two distribution:

$$\mu = 0.08$$

$$\sigma_{mix} = \sqrt{\pi^2\sigma_1^2 + (1-\pi)^2\sigma_2^2 + 2\pi(1-\pi)\sigma_1\sigma_2}$$

```
find_VaR <- function(pi){
  sigma_mix <-  sqrt(pi^2 * sigma1^2 + (1-pi)^2 * sigma2^2 + 2*pi*(1-pi)*sigma1*sigma2)
  VaR_mix <- qnorm(0.01, mu * w0, sigma_mix * w0)
  VaR_mix <- abs(VaR_mix - mu)
  return(VaR_mix)
}

pi <- 0.7

cat("Combined view VaR: ", find_VaR(pi))
```

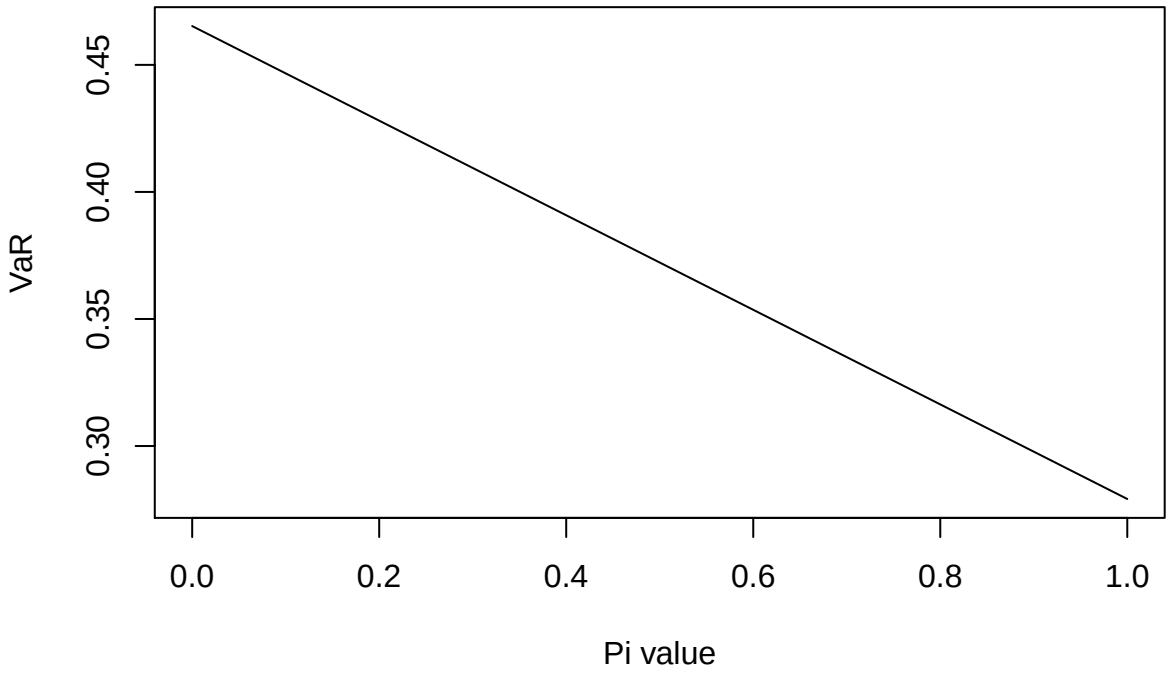
```
## Combined view VaR:  0.3349941
```

By combining the two views, we get a sigma that is smaller than both σ_1 and σ_2 Hence, the VaR we obtained for the combined distribution is also smaller than the previous two views

2.

```
pi_vec <- seq(0, 1, 0.05)

VaR_values <- c()
for(i in pi_vec){
  VaR <- find_VaR(i)
  VaR_values <- rbind(VaR_values, VaR)
```

```
}
plot(x=pi_vec, y = VaR_values, type="l", xlab="Pi value", ylab = "VaR")
```



As pi increases, the sigma decreases, and the VaR decreases.

3.

Below is a function to calculate the VaR under gamma distribution: step 1: Generate gamma distribution with appropriate alpha and beta step 2: Calculating a normal VaR given that sigma step 3: Take the quantile of the distribution of the VaR in step 2.

```
# alpha: parameter for gamma distribution
# beta: parameter for gamma distribution
# n: number of random variables to be simualted
# w0: w0 of the position
# q: bottom point. e.g. 1%
VaR_Gamma_Dist <- function(alpha, beta, n, w0, q){
  set.seed(4444)
  gammaSigmalist <- rgamma(n,alpha,beta)

  VaR_values <- rep(0,n)

  for(i in 1:n){
    curSigma <- gammaSigmalist[i]
    VaR_gamma <- qnorm(q, mu * w0, curSigma * w0)
    VaR_values[i] <- abs(mu - VaR_gamma)
  }

  return(quantile(VaR_values,q))
}
```

Qn 3 VaR for options

1.

Formula:$prob(s < S_0 - VaR) = 1 - c$

CDF of lognormal is

$$F(x) = \Phi(\frac{lnx - \mu}{\sigma})$$

Inverse is: $F^-1(1-c) = e^{\hat{\sigma}\Phi^{-1}(1-c)+\hat{\mu}}$ Where $\hat{\mu} = logS0 + (\mu - 0.5\sigma^2)t$ $\hat{\sigma} = \sigma\sqrt{t}$

$$VaR = S0(1 - exp^{\sigma\sqrt{t}\Phi^-1(1-c)+(\mu-0.5\sigma^2)t})$$

In this case, VaR is:

```
S0 <- 50
r <- 0.02
mu <- 0.07
sigma <- 0.16
t <- 252
c <- 0.99

VaR_stock <- 50*(1- exp(sigma* sqrt(10/t) * qnorm(1-c) + (mu - 0.5*sigma^2) * 10/t))
VaR_stock
```

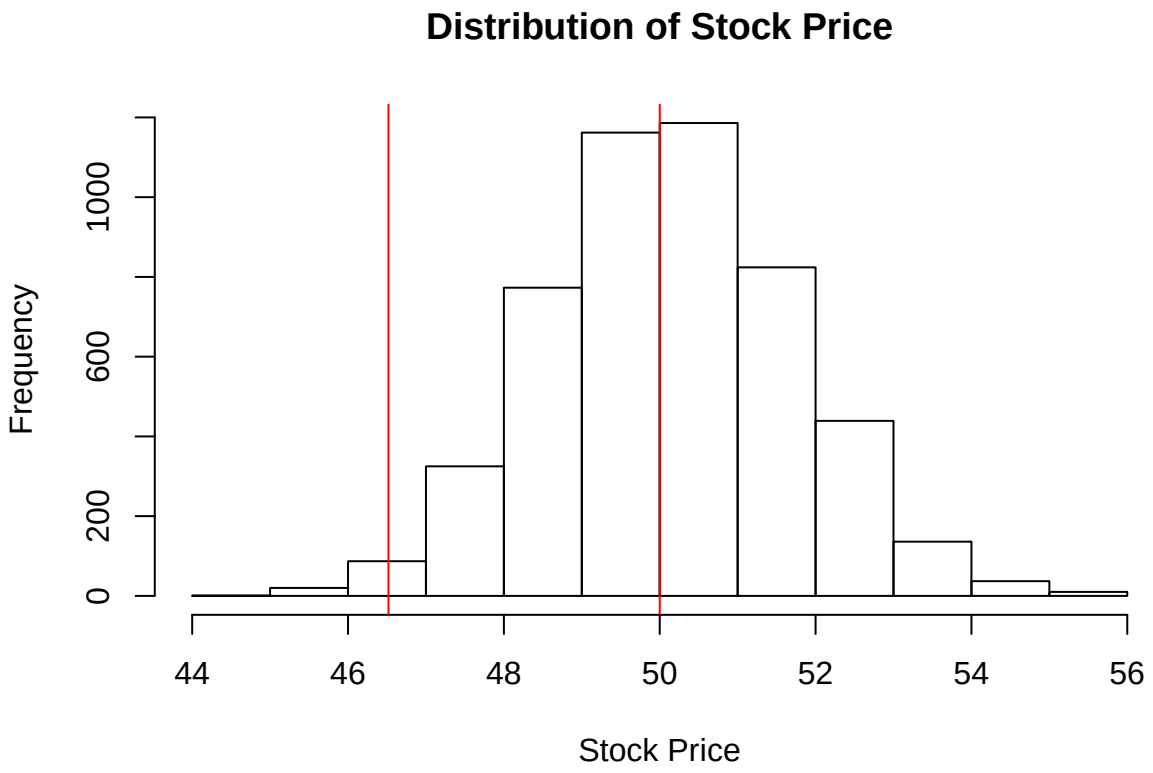
```
## [1] 3.467746
```

The distribution could also be obtained using Monte Carlo simulation

We can generate 5000 stock process and take the 1% quantile of the distribution The distribution looks like this:

```
GeneratePricePath <- function(nPath, nSims, S, r, sigma, delta){
  # generate two sets of standard normals that are negatively correlated
  set.seed(12345)
  stdNor1 <- matrix(rnorm(nPath*nSims/2), nrow = nPath/2, byrow = TRUE)
  stdNor2 <- -stdNor1
  stdNor_total <- rbind(stdNor1,stdNor2)

  priceProcess <- matrix(0, nPath, nSims+1)
  priceProcess[,1] <- S

  for(i in 1:nSims){
    priceProcess[, i+1] <- priceProcess[, i] *
    exp((r - 0.5 * sigma * sigma)
    * delta + sigma * stdNor_total[,i] * sqrt(delta))
  }
  return(priceProcess)
}

nPath <- 5000
nSims <- 10
S0 <- 50
mu <- 0.07
sigma <- 0.16
t <- 10/252
delta <- t/nSims
```

```r
stock_price <- GeneratePricePath(nPath, nSims, S0, mu, sigma, delta)

stock_dist <- stock_price[,nSims+1]

stock_dist <- sort(stock_dist, decreasing = FALSE)

value_1_percent <- quantile(stock_dist, probs=0.01)
hist(stock_dist, main="Distribution of Stock Price", xlab= "Stock Price")
abline(v= value_1_percent, col ="red")
abline(v= S0, col ="red")
```



VaR is the distance between S0 and 1% cut off, which is 3.48

2.

```r
totalCap <- 100
r <- 0.02
t <- 10/252

# step 1: find the number of shares subject to the constraint that VaR < 100mil
num_share <- totalCap/VaR_stock

# step 2: find out the total value for the long position for shares
#and short position for bond
longPosShare <- num_share * 50
shortPosShare <- longPosShare - totalCap

# step 3: find weights on bond and stock
w_share <- longPosShare/totalCap
w_bond <- shortPosShare/totalCap
```

```r
# step 4: find expected stock return and find the average return with such a portfolio
expectedRetStock <- exp(mu * 10/252) -1
avg_return <- (longPosShare * expectedRetStock - shortPosShare * (r * t))/totalCap

cat("Long stock weight is ", w_share,"\n")
```

```
## Long stock weight is  14.41859
```

```r
cat("Short bond weight is ", w_bond,"\n")
```

```
## Short bond weight is  13.41859
```

```r
cat("Average return is ",avg_return)
```

```
## Average return is  0.02945764
```

3.

```r
r <- 0.02
t <- 0.25
s0 <- 50
mu <- 0.07
sigma <- 0.16
x <- 50
nPath <- 5000
nSims <- 10
delta <- (10/252)/nSims


# function for black-schole call
bsPriceCall <- function(r, sigma, t, s0, x){
   d1 <- (log(s0/x) + (r+ 0.5*sigma*sigma) * t)/(sigma*sqrt(t))
   d2 <- d1 - sigma * sqrt(t)
   return(s0 * pnorm(d1) - (x * pnorm(d2)/exp(r* t)))
}

# function for black-schole put
bsPricePut <- function(r, sigma, t, s0, x){
  d1 <- (log(s0/x) + (r+ 0.5*sigma*sigma) * t)/(sigma*sqrt(t))
  d2 <- d1 - sigma * sqrt(t)
  return(x * pnorm(-d2)/exp(r*t) - s0 * pnorm(-d1));
}

# step 1:find out the option price using bs formula
callPrice <- bsPriceCall(r, sigma, t, s0, x)

# step 2: generate the stock price
stock_price <- GeneratePricePath(nPath, nSims, s0, mu, sigma, delta)

# step 3: generate stock price at t and find the option price distribution
stock_price_t <- bsPriceCall(r, sigma, t -(10/252), stock_price[,11], x)

# step 4: option price
VaR_Call <- callPrice - quantile(stock_price_t, 0.01)

cat("Call VaR is : ", VaR_Call,"\n")
```

```
## Call VaR is :  1.381297
```

```
# step 1: find the number of shares subject to the constraint that VaR < 100mil
num_option <- totalCap/VaR_Call

# step 2: find out the total value for the long position for Call Options
#and short position for bond
longPosOption <- num_option * callPrice
shortPosBond <- longPosOption - totalCap

# step 3: find weights on bond and stock
w_option <- longPosOption/totalCap
w_bond <- shortPosBond/totalCap

cat("Long option weight is ", w_option,"\n")
```

```
## Long option weight is  1.244597
```

```
cat("Short bond weight is ", w_bond,"\n")
```

```
## Short bond weight is  0.2445974
```

4.

```
# step 1:find out the option price using bs formula
putPrice <- bsPricePut(r, sigma, t, s0, x)

# step 2: generate stock price at t and find the option price distribution
stock_price_t <- bsPricePut(r, sigma, t -(10/252), stock_price[,11], x)

# step 3: option price
VaR_Put <-  quantile(stock_price_t, 0.99) - putPrice

cat("Put VaR is : ", VaR_Put,"\n")
```

```
## Put VaR is :  2.139182
```

```
# step 1: find the number of shares subject to the constraint that VaR < 100mil
num_option <- totalCap/VaR_Put

# step 2: find out the total value for the long position for Call Options
#and short position for bond
shortPosOption <- num_option * putPrice
longPosBond <- totalCap + shortPosOption

# step 3: find weights on bond and stock
w_option <- shortPosOption/totalCap
w_bond <- longPosBond/totalCap

cat("short option weight is ", w_option,"\n")
```

```
## short option weight is  0.6870769
```

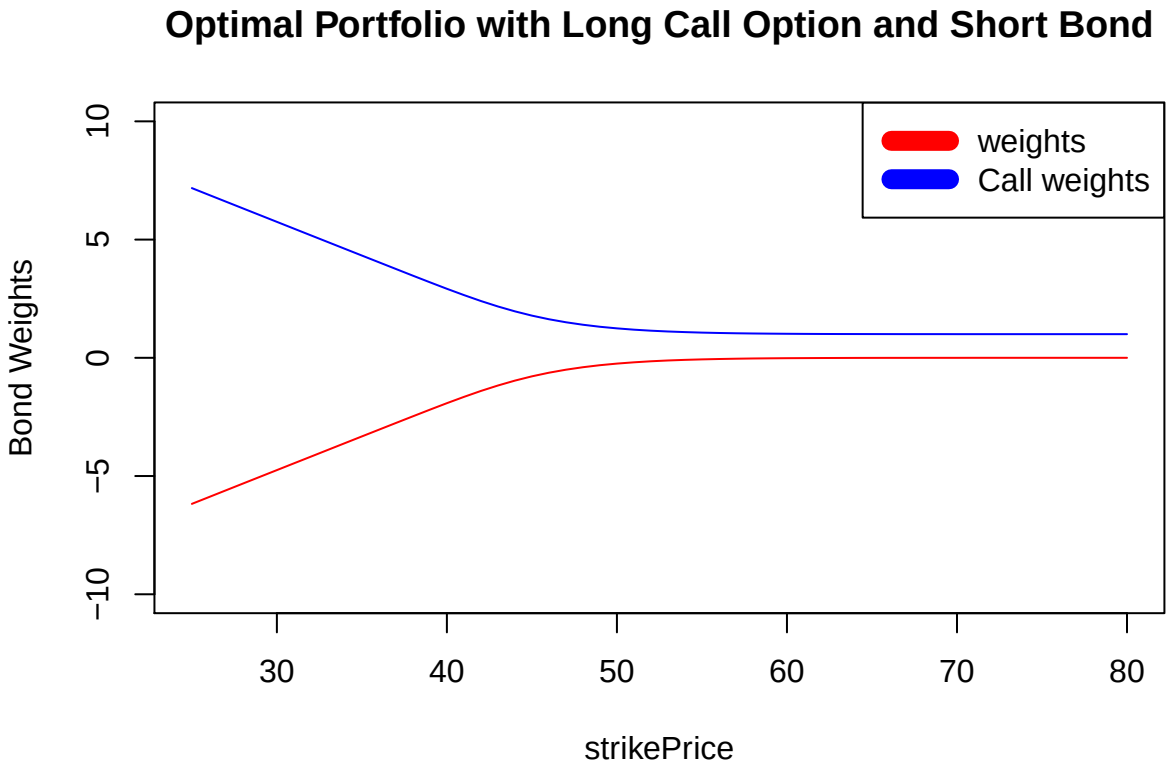
```
cat("long bond weight is ", w_bond,"\n")
```

```
## long bond weight is  1.687077
```

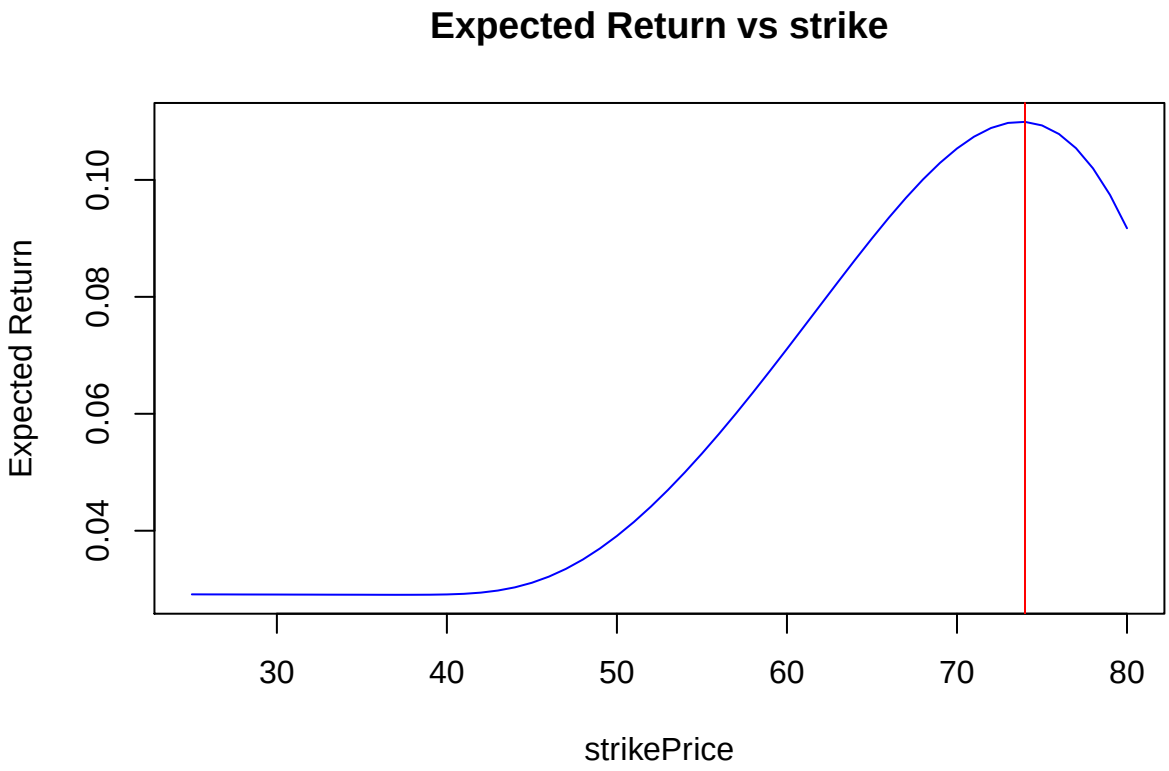
5.

Choose long call option and short bond with strike price range from 25 to 75

```
r <- 0.02
t <- 0.25
s0 <- 50
mu <- 0.07
sigma <- 0.16
x <- 50
nPath <- 5000
nSims <- 10
delta <- (10/252)/nSims
strikePrice <- seq(25,80,1)
w_option <- c()
w_bond <- c()
expected_ret <- c()
totalCap <- 100

# for call
for(i in strikePrice){
  callPrice <- bsPriceCall(r, sigma, t, s0, i)
  stock_price <- GeneratePricePath(nPath, nSims, s0, mu, sigma, delta)
  stock_price_t <- bsPriceCall(r, sigma, t -(10/252), stock_price[,11], i)
  VaR_Call <- callPrice - quantile(stock_price_t, 0.01)

  num_option <- totalCap/VaR_Call

  longPosOption <- num_option * callPrice
  shortPosBond <- longPosOption - totalCap

  w_option <- rbind(w_option,longPosOption/totalCap)
  w_bond <- rbind(w_bond, shortPosBond/totalCap)

  callPriceAfter10d <- mean(stock_price_t)

  ret <- (num_option * (callPriceAfter10d - callPrice) - shortPosBond * r * (10/252))/totalCap
  expected_ret <- rbind(expected_ret, ret)
}

plot(x= strikePrice, y = w_bond * -1,col="red", type ="l", ylab="Bond Weights",ylim = c(-10,10), main =
lines(x= strikePrice, y = w_option, col="blue",type = "l")
legend("topright", c("weights", "Call weights"), col=c("red", "blue"), lwd=10)
```




```
plot(x= strikePrice, y = expected_ret ,col="blue", type ="l", ylab="Expected Return", main = "Expected
abline(v=74,col="red")
```



As the strike price increases, the call is out of the money hence become less expensive, thus the weights on the call option and bond decrases accrodingly as the strike price increases.

I would choose strike price = 74 to obtain a optimal portfolio based on the graph.

6.

Expected Shortfall would mean that you are calculating the expected loss at 1% case, which is greater or equal to VaR. Putting this constraint would mean that the portfolio position could be more leveraged.