Project 2

Lorenzo Ausiello, Stephan Bilyk, Fabrizio Dimino

2023-12-19

PROBLEM 1

```
##Problem 1

#Setting the directory
setwd("C:/Users/loaus/OneDrive - stevens.edu/STEVENS/Intro to Financial Risk Management/Project/Project2")

#Downloading data from csv file
tbond.data<-read.csv('bonds.csv')
head(tbond.data)
```

```
##   Bond.Number Coupon  Price Yield Maturity Period
## 1           1   2.75  99.74  2.88        2      1
## 2           2   2.88  99.11  3.07        5      1
## 3           3   2.88  97.00  3.23       10      1
## 4           4   3.00  92.47  3.40       30      1
## 5           5   1.50 100.06  1.47        2      2
## 6           6   1.50 100.50  1.40        5      2
```

```
#Downloading Treasury Yields from FRED Database
getSymbols(c("DGS2", "DGS5", "DGS10", "DGS30"), src = 'FRED')
```

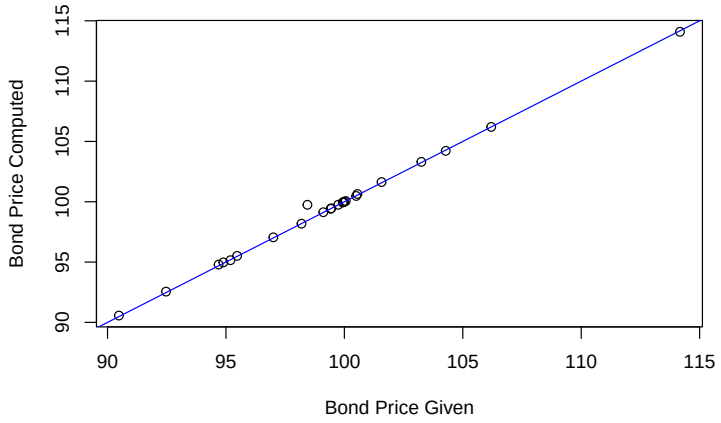
```
## [1] "DGS2"  "DGS5"  "DGS10" "DGS30"
```

```
#Merging Treasury Yields and removing missing values
t.yields <- na.omit(merge(DGS2, DGS5, DGS10, DGS30))
t.yields <- t.yields["2018-01-02/2022-03-31"]

## 1.1
# Retrieving dates
tol <- 0.02
period1 <- t.yields[abs(t.yields$DGS2 - 2.88) <= tol & abs(t.yields$DGS5 - 3.07) <= tol
                    & abs(t.yields$DGS10 - 3.23) <= tol & abs(t.yields$DGS30 - 3.40) <= tol]

tol <- 0.025
period2 <- t.yields[abs(t.yields$DGS2 - 1.47) <= tol & abs(t.yields$DGS5 - 1.40) <= tol
                    & abs(t.yields$DGS10 - 1.56) <= tol & abs(t.yields$DGS30 - 2.06) <= tol]

tol <- 0.05
```

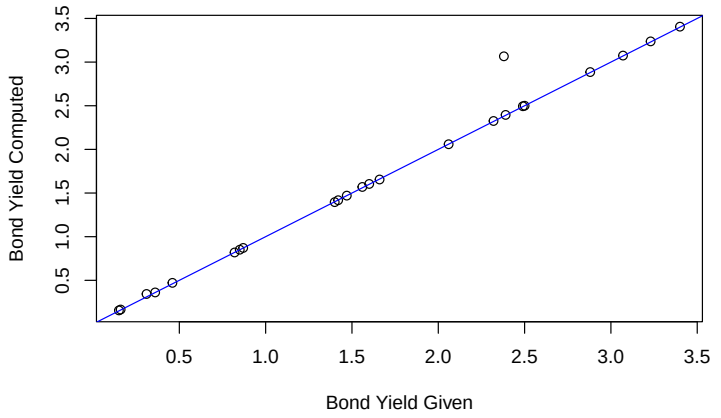
```r
period3 <- t.yields[abs(t.yields$DGS2 - 0.31) <= tol & abs(t.yields$DGS5 - 0.46) <= tol
                    & abs(t.yields$DGS10 - 0.85) <= tol & abs(t.yields$DGS30 - 1.42) <= tol]

tol <- 0.015
period4 <- t.yields[abs(t.yields$DGS2 - 0.15) <= tol & abs(t.yields$DGS5 - 0.36) <= tol
                    & abs(t.yields$DGS10 - 0.82) <= tol & abs(t.yields$DGS30 - 1.60) <= tol]
period4 <- period4[3]

tol <- 0.015
period5 <- t.yields[abs(t.yields$DGS2 - 0.16) <= tol & abs(t.yields$DGS5 - 0.87) <= tol
                    & abs(t.yields$DGS10 - 1.66) <= tol & abs(t.yields$DGS30 - 2.32) <= tol]
period5 <- period5[1]

tol <- 0.05
period6 <- t.yields[abs(t.yields$DGS2 - 2.38) <= tol & abs(t.yields$DGS5 - 2.50) <= tol
                    & abs(t.yields$DGS10 - 2.39) <= tol & abs(t.yields$DGS30 - 2.49) <= tol]

rbind(period1, period2, period3, period4, period5, period6)
```

```
##            DGS2 DGS5 DGS10 DGS30
## 2018-10-05 2.88 3.07  3.23  3.40
## 2019-10-07 1.46 1.38  1.56  2.05
## 2020-03-26 0.30 0.51  0.83  1.42
## 2020-11-06 0.16 0.36  0.83  1.60
## 2021-04-06 0.16 0.88  1.67  2.32
## 2022-03-29 2.35 2.49  2.41  2.53
```

```r
## 1.2
# Calculating prices and comparing with the given ones
bondprice <- function(Coupon, Yield, Maturity, Face){
  Yield <- Yield/100
  Coupon <- Coupon/100
  Coupon/Yield*Face*(1-1/((1+Yield)^Maturity))+Face/((1+Yield)^Maturity)}

Face <- 100
tbond.data$Bond_Price <- apply(tbond.data, 1, function(row) {
  bondprice(row["Coupon"], row["Yield"], row["Maturity"], Face)
})


{plot(tbond.data$Price, tbond.data$Bond_Price, xlab="Bond Price Given", ylab = "Bond Price Computed")
  abline(0, 1, col = "blue")}
```



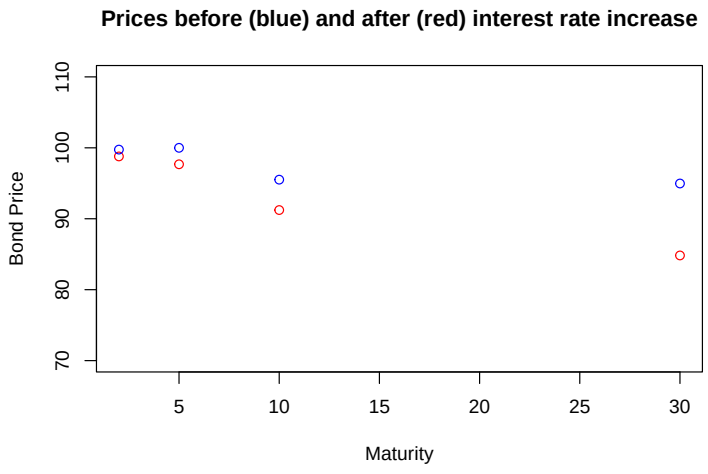

```r
## 1.3
# Calculating yields and comparing with the given ones

bondyield <- function(yield, Coupon, Price, Maturity, Face){
  Coupon <- Coupon/100
  Bprice <- Coupon/yield*Face*(1-1/((1+yield)^Maturity))+Face/((1+yield)^Maturity) - Price
  return(Bprice)
}

tbond.data$Bond_Yield <- apply(tbond.data, 1, function(row) {
  Coupon <- row["Coupon"]
  Price <- row["Price"]
  Maturity <- row["Maturity"]
  Face <- 100
  uniroot(function(y) bondyield(y, Coupon, Price, Maturity, Face), c(0.0001, 1))$root
})*100

{plot(tbond.data$Yield, tbond.data$Bond_Yield, xlab = "Bond Yield Given", ylab = "Bond Yield Computed")
  abline(0, 1, col = "blue")}
```



```r
## 1.4
# Compute Macaulay duration
mcduration <- function(Coupon, Bond_Price, Yield, Maturity){
  w <- c()
  for (i in 1:Maturity){
    w[i] <- Coupon/((1+Yield/100)^i)/Bond_Price}
  w[Maturity] <- w[Maturity]+(100/((1+Yield/100)^Maturity))/Bond_Price
  Duration <- sum(w*(1:Maturity))
  return(Duration)}

tbond.data$Bond_Duration <- apply(tbond.data, 1, function(row) {
  mcduration(row["Coupon"], row["Bond_Price"],row["Yield"], row["Maturity"])
})

summaryBy(Bond_Duration ~ Maturity, data = tbond.data, FUN = c(min,max))
```

```
##   Maturity Bond_Duration.min Bond_Duration.max
## 1        2          1.973203          1.998701
## 2        5          4.726402          4.975043
## 3       10          8.808622          9.719726
## 4       30         19.758713         24.593082
```

```r
## 1.5
# calculate the change in the Treasury bond prices Using first order Taylor expansion
newprices <- c()
for (i in 21:24){
  newprices[i-20] <- tbond.data$Bond_Price[i]-0.005*tbond.data$Bond_Duration[i]/(1+tbond.data$Yield[i]/

{plot(tbond.data$Maturity[21:24], newprices, ylim = c(70,110), col = 'red',
      xlab = "Maturity", ylab = "Bond Price", main = "Prices before (blue) and after (red) interest rat
points(tbond.data$Maturity[21:24], tbond.data$Bond_Price[21:24], col = 'blue')}
```




```r
# compare your Taylor expansion results with the exact price
apply(tbond.data, 1, function(row) {
  bondprice(row["Coupon"], row["Yield"]+0.50, row["Maturity"], Face)})[21:24]==newprices
```

```
## [1] FALSE FALSE FALSE FALSE
```

```
apply(tbond.data, 1, function(row) {
  bondprice(row["Coupon"], row["Yield"]+0.50, row["Maturity"], Face)})[21:24]-newprices
```

```
## [1] 0.006987125 0.033060654 0.109669513 0.650451933
```

```
## 1.6
# how many units of each bond to have a portfolio duration equal to 4?
w21 <- (4-tbond.data$Bond_Duration[22])/(tbond.data$Bond_Duration[21]-tbond.data$Bond_Duration[22])
w22 <- (1-w21)
duration.tot <- w21*tbond.data$Bond_Duration[21] + w22*tbond.data$Bond_Duration[22]
n22 <- w22*100000/tbond.data$Price[22]
n21<- w21*100000/tbond.data$Price[21]
data.frame(Bond21=c(w21,n21), Bond22=c(w22,n22), row.names = c("weights", "units"))
```

```
##             Bond21     Bond22
## weights   0.273697   0.726303
## units   278.034333 726.303003
```

```
duration.tot
```

```
## [1] 4
```

```
# how many units of each bond to have a portfolio duration equal to 8?
w21 <- (8-tbond.data$Bond_Duration[22])/(tbond.data$Bond_Duration[21]-tbond.data$Bond_Duration[22])
w22 <- (1-w21)
duration.tot <- w21*tbond.data$Bond_Duration[21] + w22*tbond.data$Bond_Duration[22]
n22 <- w22*100000/tbond.data$Price[22]
n21<- w21*100000/tbond.data$Price[21]
data.frame(Bond21=c(w21,n21), Bond22=c(w22,n22), row.names = c("weights", "units"))
```

```
##               Bond21      Bond22
## weights    -1.163081    2.163081
## units   -1181.512880 2163.081279
```

```
duration.tot
```

```
## [1] 8
```

```
## 1.7: Bonus question
# how many units of each bond to have a portfolio duration equal to 8?
d <- tbond.data$Bond_Duration[21:24]
d.tot <- 8

findVector <- function(d, d.tot) {
  objective <- function(X) {
    diff <- sum(X * d) - d.tot
    penalty <- (sum(X) - 1)^2
    return(diff^2 + penalty)
```

```
  }

  set.seed(123)
  X <- runif(length(d))
  result <- optim(X, objective, method = "L-BFGS-B", lower = rep(0, length(d)), upper = rep(1, length(d
  return(result$par)
}


w <- findVector(d, d.tot)
data.frame(weights=w, units=(w * 100000)/tbond.data$Price[21:24],
           row.names = c("Bond21", "Bond22", "Bond23", "Bond24"))
```

```
##          weights    units
## Bond21 0.2408808 244.6980
## Bond22 0.3955957 395.5957
## Bond23 0.1825058 191.1656
## Bond24 0.1810177 190.7659
```

```
w%*%tbond.data$Bond_Duration[21:24]
```

```
##      [,1]
## [1,]    8
```

Question 1.4 In the columns we notice that bonds with longer maturities show obviously higher durations: final cash flows (face value) are paid further in time and their present value has a greater impact on average duration. In the rows we note min and max range that is given by the different characteristics of the bonds (such as different coupons).‘ The higher the maturity, the wider the range: this is due to the fact that for increasing maturity, even a small change in coupon rate implies change in many more cash flows, impacting heavily on the weights and therefore durations.

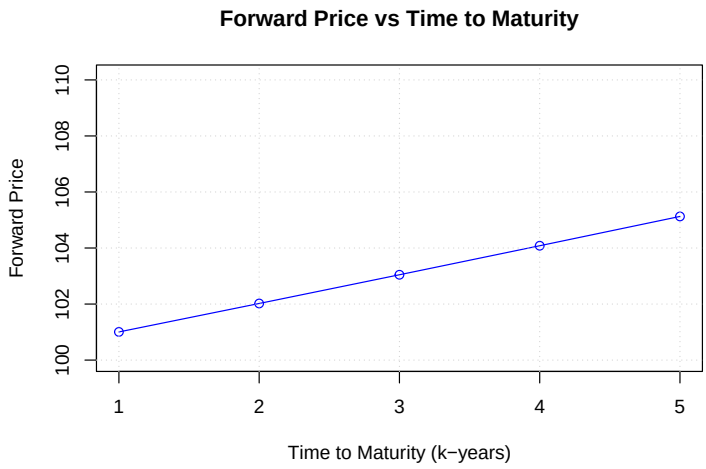
Question 1.5 We can notice that when interest rates go up, prices go down. However the variation is not always the same: duration (and modified duration that is proportional) is a measure of exposure to the interest rates risk/volatility. The higher the duration, the higher the bond price variation when interest rates vary. The same we can state for maturity: when maturity is higher, all other conditions being equal, the duration will be higher: the face value (the most important cash flow) and the other cash flows are paid further in time and therefore they are discounted for more years before the price incorporates them. When yields will change the discount factor and the PV of the cash flows will be more affected causing a higher bond price variation .

Question 1.6 When you need to limit your portfolio duration to 4 years using only Bond 21 and Bond 22, the portfolio should consist of 27.37% allocation in Bond 21 and 72.63% allocation in Bond 22. This is equivalent to 278 units of Bond 21 and 726 units of Bond 22. The total duration of the portfolio is effectively 4 years. Bond 21 has a duration equal to almost 2 and Bond 22 has a duration equal to almost 4.75. Therefore, to achieve a target duration of 4 years, we should allocate most of the budget to Bond 22 (that present a duration just slightly higher than the target) and less than 30% to bond 21 to lower the overall portfolio duration until 4.

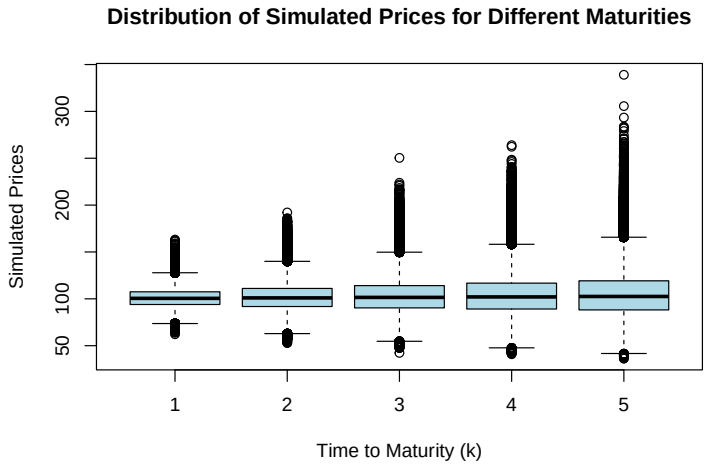
When your portfolio duration target is equal to 8 using only Bond 21 and Bond 22, you need to short -1181.5 units of Bond 21, while investing in bond 22 bringing the units into 2163.08. Both Bond 21 and Bond 22 present a duration lower than the target: therefore you need to short the bond with the lowest duration (21) and exploit the leverage effect to allocate the capital. As a result you will get a duration equal to the duration of bond with the highest duration (22) + the difference between the duration of bond 22 minus duration of bond 21 (scaled for the % of portfolio shorted).

PROBLEM 2

```r
## Problem 2
r <- 0.01
sigma <- 0.1
d <- 0
S0 <- 100
k <- 1:5

## 2.1
# no-arbitrage pricing: fair value of a k-years forward contract
forward_prices <- S0 * exp((r-d) * k)

{plot(k, forward_prices, type = 'o', col = 'blue',
     xlab = 'Time to Maturity (k-years)', ylab = 'Forward Price',
     main = 'Forward Price vs Time to Maturity',
     xlim = c(1, 5), ylim = c(100, max(forward_prices) + 5))
grid()}
```




```r
## 2.2
# Montecarlo simulations pricing
n <- 10^6

GBM <- function(k) {
  future_price <- S0 * exp((r - 0.5 * sigma^2) * k + sigma * sqrt(k) * rnorm(n))
  return(future_price)
}

simulated_prices <- matrix(NA, nrow = n, ncol = length(k))
for (j in 1:length(k)) {
    simulated_prices[,j] <- GBM(k[j])}

# distribution of the simulated prices
boxplot(simulated_prices, col = "lightblue", xlab = "Time to Maturity (k)",
        ylab = "Simulated Prices", main = "Distribution of Simulated Prices for Different Maturities")
```




```r
# forward prices
data.frame(Expected_future_price=apply(simulated_prices, 2, FUN = mean),
           row.names=c("k=1","k=2","k=3","k=4","k=5"))
```

```
##     Expected_future_price
## k=1              100.9997
## k=2              101.9972
## k=3              103.0458
## k=4              104.0872
## k=5              105.1058
```

```r
## 2.3 a)
# Strategy A: forward contract
# at t0 long forward, forward price that you believe
# will be lower than S1 (index price after 1 year)
# at t1 (after 1 year) buy (-K) and sell (+S1) the stock

PnL_forward <- simulated_prices[,1] - forward_prices[1]

## 2.3 b)
# Strategy B: underlying asset
# at t0 you borrow money (+S0) at risk free rate,
# at t0 you buy the stock index (-S0)
# after 1 year (t1) you sell the stock (+S1)
# and give back the money to the lender (-S0 * exp(r * 1)=-K)

PnL_underlying <- simulated_prices[,1] - (S0 * exp(r * 1))

## 2.3 c)
# PnL
expected_PnL_forward <- mean(PnL_forward)
VaR_forward <- expected_PnL_forward - quantile(PnL_forward, 0.05)

expected_PnL_underlying <- mean(PnL_underlying)
VaR_underlying <- expected_PnL_forward - quantile(PnL_underlying, 0.05)
data.frame(StrategyA=c(expected_PnL_forward, VaR_forward),
```

```
          StrategyB=c(expected_PnL_underlying,VaR_underlying),
          row.names=c("Expected PnL", "VaR 95% PnL"))
```

```
##                 StrategyA    StrategyB
## Expected PnL -0.005337412 -0.005337412
## VaR 95% PnL  15.747748827 15.747748827
```

Question 2.1 As the maturity increases, the price of the forward increases, due to the potential upward movement of the underlying index. The higher the maturity, the higher the "t" in the forward price formula.

Question 2.2 The no-arbitrage price is consistent with risk-neutral estimation under market efficiency assumptions. Therefore, risk-neutral Monte Carlo simulations lead to forward prices estimations similar to the previous no-arbitrage prices estimations. The more the simulations, the less the difference between the two. The box plot shows that for increasing maturity the future price range (max-min) becomes wider. Comparisons between graph in 2.1 and box plot make sense: the higher the maximum value that the price can assume in the future, the higher the forward price (fair fixed future price).

PROBLEM 3

```
## Problem 3

# retrieve quotes for the GBP/USD forward contracts
S0 <- 1.2273
Q <- (read.csv('FE535_Forward_Prices.csv')[['Ask']]+read.csv('FE535_Forward_Prices.csv')[['Bid']])/2
F0 <- S0 + Q/10^4
names(F0) <- read.csv('FE535_Forward_Prices.csv')[['Name']]
F0
```

```
##  GBPUSD 1M FWD  GBPUSD 2M FWD  GBPUSD 3M FWD  GBPUSD 4M FWD  GBPUSD 5M FWD
##       1.227505       1.227888       1.228134       1.228386       1.228718
##  GBPUSD 6M FWD  GBPUSD 7M FWD  GBPUSD 8M FWD  GBPUSD 9M FWD GBPUSD 10M FWD
##       1.228967       1.229260       1.229514       1.229729       1.229976
## GBPUSD 11M FWD  GBPUSD 1Y FWD
##       1.230222       1.230434
```

```
## 3.1 a)
#Theta forward-looking
theta <- log(F0/S0)*12/1:12
theta
```

```
##  GBPUSD 1M FWD  GBPUSD 2M FWD  GBPUSD 3M FWD  GBPUSD 4M FWD  GBPUSD 5M FWD
##    0.002004233    0.002873914    0.002717239    0.002653434    0.002771316
##  GBPUSD 6M FWD  GBPUSD 7M FWD  GBPUSD 8M FWD  GBPUSD 9M FWD GBPUSD 10M FWD
##    0.002714689    0.002735533    0.002702892    0.002636247    0.002613627
## GBPUSD 11M FWD  GBPUSD 1Y FWD
##    0.002594632    0.002550318
```

```
## 3.1 b)
#Theta from Libor
libor <- read.csv('FE535_Libor_USD_GBP.csv')
```

```r
libor <- libor[libor$Dates=='11/13/2023',]

theta1M <- libor['US0001M.Index']/100-libor['BP0001M.Index']/100
theta3M <- libor['US0003M.Index']/100-libor['BP0003M.Index']/100
theta6M <- libor['US0006M.Index']/100-libor['BP0006M.Index']/100
c(theta1M, theta3M, theta6M)
```

```
## $US0001M.Index
## [1] 0.0123548
##
## $US0003M.Index
## [1] 0.0028025
##
## $US0006M.Index
## [1] 0.0109803
```

```r
# How does calibrated theta compare with LIBOR rates?
theta.compare <- matrix(c(theta1M, theta3M, theta6M, theta[1], theta[3], theta[6]), ncol=2)
rownames(theta.compare)=c('Theta 1M', 'Theta 3M', 'Theta 6M')
colnames(theta.compare)=c('Libor','Forward-Looking Approach')
theta.compare
```

```
##          Libor     Forward-Looking Approach
## Theta 1M 0.0123548 0.002004233
## Theta 3M 0.0028025 0.002717239
## Theta 6M 0.0109803 0.002714689
```

```r
## 3.1 c)
#calibrating sigma
symbol <- "GBPUSD=X"
start_date <- "2018-01-01"
end_date <- "2022-04-03"

GBPUSD <- getSymbols(symbol, from = start_date, to = end_date, src = "yahoo", auto.assign = F)
GBPUSD <- na.omit(GBPUSD)
returns <- na.omit(log(GBPUSD$`GBPUSD=X.Adjusted`/lag(GBPUSD$`GBPUSD=X.Adjusted`)))
head(returns)
```

```
##            GBPUSD=X.Adjusted
## 2018-01-02     -0.0003513551
## 2018-01-03      0.0062618301
## 2018-01-04     -0.0062618301
## 2018-01-05      0.0030717406
## 2018-01-08      0.0016956344
## 2018-01-09     -0.0003394415
```

```r
sigma <- sd(returns) *  sqrt(252)
sigma
```

```
## [1] 0.08552515
```

```r
## 3.2
#VaR for the Unhedged
GBM <- function(n) {
  ST <- S0 * exp((theta[11] - 0.5 * sigma^2) * 11/12 + sigma * sqrt(11/12) * rnorm(n))
  return(ST)
}

ST <- GBM(10^6)

Vt <- 1.25*10^6*(ST - S0)

Expected_Vt <- mean(Vt)
VaR_Vt <- Expected_Vt - quantile(Vt, 0.01)
data.frame(PnL=c(Expected_Vt, VaR_Vt), row.names=c("Expected PnL", "VaR 99% PnL"))
```

```
##                       PnL
## Expected PnL   3537.882
## VaR 99% PnL  271179.946
```

```r
## 3.3
#Unitary Hedge
units <- 62500
n.contracts <- 20
quantity <- units * n.contracts

## 3.3 a)
#Dec2024
F0 <- as.numeric(getSymbols("6BZ24.CME", src = "yahoo", auto.assign = F)["2023-11-13"][,"6BZ24.CME.Adjus
S1 <- ST
F1 <- ST*exp(theta[2]*2/12)
PnL <- quantity * (S1 - F1) - quantity * (S0 - F0)
Expected_PnL <- mean(PnL)
VaR_PnL <- Expected_PnL - quantile(PnL, 0.01)
data.frame(PnL=c(Expected_PnL, VaR_PnL), row.names=c("Expected PnL", "VaR 99% PnL"))
```

```
##                     PnL
## Expected PnL 4013.2590
## VaR 99% PnL   151.8897
```

```r
## 3.3 b)
#Sep2024
GBM <- function(n) {
  ST <- S0 * exp((theta[10] - 0.5 * sigma^2) * 10/12 + sigma * sqrt(10/12) * rnorm(n))
  return(ST)
}

ST <- GBM(10^6)
F0 <- as.numeric(getSymbols("6BU24.CME", src = "yahoo", auto.assign = F)["2023-11-13"][,"6BU24.CME.Adjus
S1 <- ST

GBM <- function(n) {
  ST <- S1 * exp((theta[1] - 0.5 * sigma^2) * 1/12 + sigma * sqrt(1/12) * rnorm(n))
```

```
  return(ST)
}

S2 <- GBM(10^6)
PnL <- quantity * (S2 - S1) - quantity * (S0 - F0)
Expected_PnL <- mean(PnL)
VaR_PnL <- Expected_PnL - quantile(PnL, 0.01)
data.frame(PnL=c(Expected_PnL, VaR_PnL), row.names=c("Expected PnL", "VaR 99% PnL"))
```

```
##                     PnL
## Expected PnL  4458.698
## VaR 99% PnL  87031.208
```

```
## 3.4
# Hedging using ETFs

# ETFs:
# FXB: Invesco CurrencyShares British Pound Sterling Trust
# EUO: ProShares UltraShort Euro
# DGBP: WisdomTree Bloomberg U.S. Dollar Bullish Fund
# UDN: Invesco DB U.S. Dollar Index Bearish Fund
# UUP: Invesco DB U.S. Dollar Index Bullish Fund

start_date <- "2018-01-01"
end_date <- "2022-04-03"
FXB <- getSymbols('FXB', from = start_date, to = end_date, auto.assign = F)
EUO <- getSymbols('EUO', from = start_date, to = end_date, auto.assign = F)
DGBP <- getSymbols('DGBP', from = start_date, to = end_date, auto.assign = F)
UDN <- getSymbols('UDN', from = start_date, to = end_date, auto.assign = F)
UUP <- getSymbols('UUP', from = start_date, to = end_date, auto.assign = F)

returns.FXB <- na.omit(log(FXB$FXB.Adjusted/lag(FXB$FXB.Adjusted)))
returns.EUO <- na.omit(log(EUO$EUO.Adjusted/lag(EUO$EUO.Adjusted)))
returns.DGBP <- na.omit(log(DGBP$DGBP.Adjusted/lag(DGBP$DGBP.Adjusted)))
returns.UDN <- na.omit(log(UDN$UDN.Adjusted/lag(UDN$UDN.Adjusted)))
returns.UUP <- na.omit(log(UUP$UUP.Adjusted/lag(UUP$UUP.Adjusted)))

returns <- merge(returns, returns.EUO, returns.UUP, returns.UDN, returns.FXB, returns.DGBP, join = 'inn

model1 <- lm(GBPUSD.X.Adjusted ~ FXB.Adjusted, data = returns)
model2 <- lm(GBPUSD.X.Adjusted ~ EUO.Adjusted, data = returns)
model3 <- lm(GBPUSD.X.Adjusted ~ DGBP.Adjusted, data = returns)
model4 <- lm(GBPUSD.X.Adjusted ~ UDN.Adjusted, data = returns)
model5 <- lm(GBPUSD.X.Adjusted ~ UUP.Adjusted, data = returns)

# Hedge effectiveness: FXB
summary(model1)$r.squared
```

```
## [1] 0.2594032
```

```
# Hedge effectiveness: EUO
summary(model2)$r.squared
```

```
## [1] 0.1328316
```

```
# Hedge effectiveness: DGBP
summary(model3)$r.squared
```

```
## [1] 0.233197
```

```
# Hedge effectiveness: UDN
summary(model4)$r.squared
```

```
## [1] 0.1774785
```

```
# Hedge effectiveness: UUP
summary(model5)$r.squared
```

```
## [1] 0.1536601
```

Question 3.3.c We may notice the highest VaR when we do not use futures (unhedged position). The case without hedging is likely to have the highest VaR due to the full exposure to exchange rate fluctuations. As for unitary hedge, therefore when the exporter uses futures contracts, we observe a higher VaR when he uses September 2024 futures contracts than with the December 2024 futures contract. Using Dec 2024 futures contracts the exporter is exposed only to the basis risk, meaning to the variation of the difference between spot price and future price over the next year. Using a futures contract that expires before delivery (Sep 2024) introduces additional risk: the exporter has a full hedging until the contract expires, then, for the remaining time (until the payment is received), there is a full exposure to exchange rate fluctuations (VaR is still less than the unhedged one, since the time of exposure is shorter).

Question 3.4 FXB (Invesco CurrencyShares British Pound Sterling Trust): This ETF directly tracks the GBP/USD exchange rate, aiming to provide a positive return when the pound strengthens and a negative return when it weakens. R-squared value of 0.2594 indicates a moderate positive correlation with GBPUSD, signifying its potential effectiveness as a hedge.

EUO (ProShares UltraShort Euro): This ETF aims to provide twice the inverse daily return of the Euro Stoxx 50 Index. Since the euro and pound often have an inverse relationship, EUO might hedge GBPUSD to some extent. However, R-squared value of 0.1328 suggests a weaker correlation compared to FXB.

DGBP (WisdomTree Bloomberg U.S. Dollar Bullish Fund): This ETF invests in USD-denominated securities and aims to benefit from a rising USD. With a positive correlation to the USD, DGBP could potentially hedge GBPUSD by offsetting gains in the pound with losses in the USD. However, R-squared value of 0.2332 suggests a similar effect as FXB, with potential redundancy if both are used simultaneously.

UDN (Invesco DB U.S. Dollar Index Bearish Fund): This ETF seeks to provide short exposure to the U.S. Dollar Index, essentially betting on a weakening USD. Therefore, it could partially hedge GBPUSD by offsetting losses in the pound with gains from a weakening USD. R-squared value of 0.1775 reflects a somewhat weaker correlation compared to DGBP.

UUP (Invesco DB U.S. Dollar Index Bullish Fund): This ETF, similar to DGBP, invests in USD-denominated securities and aims to benefit from a rising USD. Its potential hedging effect on GBPUSD is analogous to DGBP, with the R-squared value of 0.1537 suggesting a slightly weaker correlation.

Economic Rationale:

FXB: Offers direct exposure to GBP/USD fluctuations, making it a straightforward hedge for the exporter's GBP receivables.

EUO: While not directly correlated to GBPUSD, its inverse relationship with the euro can provide some hedging through portfolio diversification.

DGBP: Leverages a rising USD to offset potential gains in the pound, acting as a partial hedge depending on the relative movements of both currencies.

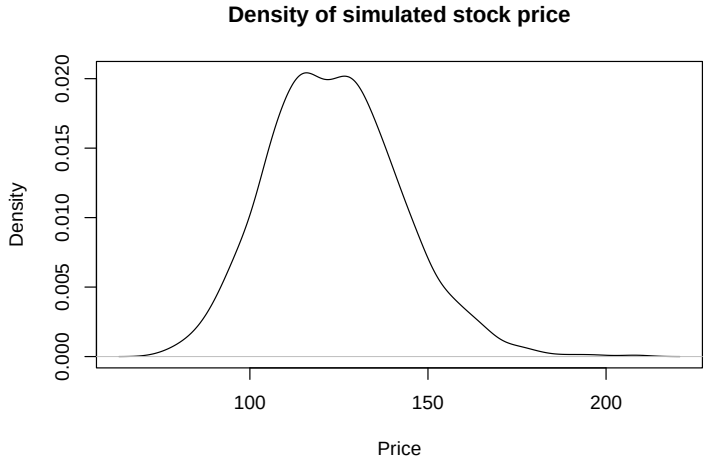
UDN: By providing short exposure to USD, UDN can partially hedge GBPUSD but comes with potential risks if the USD strengthens unexpectedly.

UUP: Like DGBP, UUP can offer partial hedging through a rising USD, but choosing between DGBP and UUP might depend on specific market expectations and risk tolerance.

In conclusion, we should remember that expense ratios of the ETFs could affect their overall hedging efficiency.

PROBLEM 4

```
## Problem 4

## 4.1.1
# Stock Price simulation: risk-neutral valuation
mu <- 0.10
sigma <- 0.15
S0 <- 120
r <- 0.03
K <- 125
t <- 1

GBM <- function(n){
  Z_seq <- rnorm(n)
  dRt_seq <- (r - sigma^2/2)*t+sigma*sqrt(t)*Z_seq
  S1 <- S0*exp(dRt_seq)
  return(list(S1, Z_seq))
}

gbm <- GBM(10^3)
S1_seq <- gbm[[1]]
Z_seq <- gbm[[2]]

plot(density(S1_seq), main = "Density of simulated stock price", xlab = "Price", ylab = "Density")
```




```
## 4.1.2
# summary statistics
S1_exp <- mean(S1_seq)
S1_var <- var(S1_seq)
cbind(S1_exp, S1_var)
```

```
##        S1_exp   S1_var
## [1,] 123.8612 345.0476
```

```
# simulations vs true-values (closed-form solutions)
s1_exp<-S0*exp(r)
s1_var<-(exp(sigma^2)-1)*S0^2*exp(2*r)

sim_vs_true<-data.frame(Mean=c(S1_exp, s1_exp), Var=c(S1_var, s1_var))
rownames(sim_vs_true)=c('Simulation', 'True Value')
sim_vs_true
```

```
##                Mean      Var
## Simulation 123.8612 345.0476
## True Value 123.6545 347.9346
```

```
## 4.2.1
# european call price based on simulations
c_seq <- pmax(0, S1_seq - K)
c <- mean(c_seq)*exp(-r*t)
c
```

```
## [1] 6.651076
```

```
## 4.2.2
# european call price based on Black-Scholes model
BS <- function(S, K, r, sigma, t){
  d2 <- (log(S/K)+((r-sigma^2/2)*t))/(sigma*sqrt(t))
  d1 <- d2 + sigma*sqrt(t)
```

```
  c <- S*pnorm(d1) - K * exp(-r*t)*pnorm(d2)
  return(c)
}

c_bsm <- BS(S0, K, r, sigma, t)
c_bsm
```

```
## [1] 6.579149
```

Question 4.2.3 The Black-Scholes model provides a closed-form solution for option pricing, while risk-neutral Monte Carlo simulations use numerical methods. Results from Black-Scholes and risk-neutral Monte Carlo simulations differ, but with a higher number of simulations the MC offer increased accuracy and stability. Therefore, with an increasing number of simulations, the prices tend to converge.

```
## 4.3
# BIAS-VARIANCE

## 4.3.1
# average option price across hundred experiments
set.seed(123)
M <- 100
GBM_c <- function(n){
  Z_seq <- rnorm(10^3)
  dRt_seq <- (r - sigma^2/2)*t+sigma*sqrt(t)*Z_seq
  S1 <- S0*exp(dRt_seq)
  c_seq <- pmax(0, S1 - K)
  c <- mean(c_seq)*exp(-r*t)
  c
  return(c)
}

c_vec <- sapply(1:M, GBM_c)
c_mean <- mean(c_vec)
c_mean
```

```
## [1] 6.588731
```

```
## 4.3.2
# MSE
squared_errors <- (c_vec - c_bsm)^2
MSE <- mean(squared_errors)
MSE
```

```
## [1] 0.12031
```

```
# decomposition MSE
bias <- mean(c_vec-c_bsm)^2
variance <- var(c_vec)
bias
```

```
## [1] 9.182407e-05
```

```r
variance
```

```
## [1] 0.1214325
```

```r
## 4.4
# Variance Reduction Approach: Antithetic Variates

## 4.4.1
# Simulations statistics
Z <- c(Z_seq[1:500], -Z_seq[1:500])
dRt_seq <- (r - sigma^2/2)*t+sigma*sqrt(t)*Z
S1_seq <- S0*exp(dRt_seq)
S1_exp <- mean(S1_seq)
S1_var <- var(S1_seq)
cbind(S1_exp, S1_var)
```

```
##        S1_exp   S1_var
## [1,] 123.5845 330.5398
```

```r
## 4.4.2
# price of the European Call option
c_seq <- pmax(0, S1_seq - K)
c <- mean(c_seq)*exp(-r*t)
c
```

```
## [1] 6.383486
```

```r
## 4.4.3
# MSE
set.seed(123)
M <- 100
GBM_c <- function(n){
  Z_seq <- rnorm(10^3)
  Z_seq <- c(Z_seq[1:500], -Z_seq[1:500])
  dRt_seq <- (r - sigma^2/2)*t+sigma*sqrt(t)*Z_seq
  S1 <- S0*exp(dRt_seq)
  c_seq <- pmax(0, S1 - K)
  c <- mean(c_seq)*exp(-r*t)
  c
  return(c)
}

c_vec <- sapply(1:M, GBM_c)
c_mean <- mean(c_vec)
c_mean
```

```
## [1] 6.535805
```

```r
squared_errors <- (c_vec - c_bsm)^2
MSE <- mean(squared_errors)
MSE
```

```
## [1] 0.0758366
```

```
# decomposition MSE

bias <- mean(c_vec-c_bsm)^2
variance <- var(c_vec)
bias
```

```
## [1] 0.001878713
```

```
variance
```

```
## [1] 0.07470494
```

Question 4.4.4

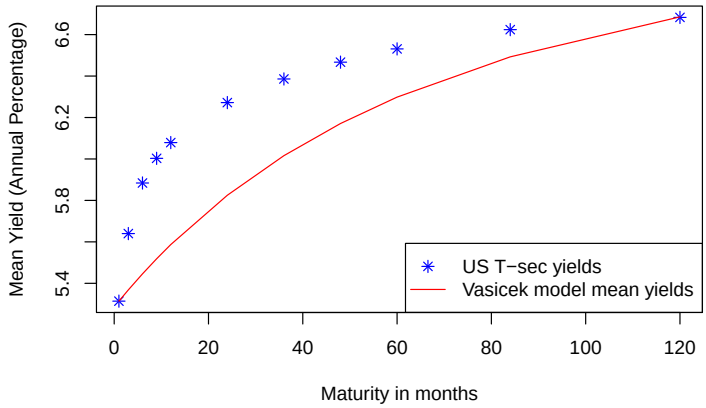
a) Yes, it did.

b) The reduction in MSE (achieved through antithetic variates) stems from a variance reduction effect. By generating paired paths negative correlated, antithetic variates exploit negative correlation to reduce variance of simulated paths. Therefore, this correlation leads to a reduction in the overall variance of the option price estimates across the hundred experiments. Consequently, since the MSE of estimates can be decomposed into the bias and the variance, the reduction of variance (and a less than proportional increase in bias) enhances the precision of the estimates, reducing overall MSE and making it an effective technique for improving the accuracy of Monte Carlo simulations in option pricing.

PROBLEM 5: BONUS

```
## Problem 5

## 5.1
# replicate Figure 1 of Backus et al. (1998)
us_gvt <- data.frame(Maturity=c(1,3,6,9,12,24,36,48,60,84,120),
                     Mean=c(5.314,5.640,5.884,6.003,6.079,6.272,6.386,6.467,6.531,6.624,6.683),
                     St.Dev=c(3.064,3.143,3.178,3.182,3.168,3.124,3.087,3.069,3.056,3.043,3.013),
                     Skewness=c(0.886,0.858,0.809,0.776,0.730,0.660,0.621,0.612,0.599,0.570,0.532),
                     Kurtosis=c(0.789,0.691,0.574,0.480,0.315,0.086,-0.066,-0.125,-0.200,-0.349,-0.477)
                     Auto=c(0.976,0.981,0.982,0.982,0.983,0.986,0.988,0.989,0.990,0.991,0.992))

theta <- us_gvt$Mean[1]/1200
varphi <- 0.976
lambda <- -0.0824
delta <- lambda^2/2
sigma <- sqrt((us_gvt$St.Dev[1]/1200)^2*(1-varphi^2))
# here the paper " Discrete-time models of bond pricing"
# wrongly report sigma=0.005560 instead of 0.0005560
B <- c()
B[1] <- 1
for (i in 2:120){
  B[i] <- 1+B[i-1]*varphi
}
```

```r
A <- c()
A[1] <- 0
for (i in 2:120){
  A[i] <- A[i-1]+delta+B[i-1]*(1-varphi)*theta-((lambda+B[i-1]*sigma)^2)/2
}
A <- A[c(1,3,6,9,12,24,36,48,60,84,120)]
B <- B[c(1,3,6,9,12,24,36,48,60,84,120)]

bond.ylds_exp <- (A+B*theta)*us_gvt$Maturity^-1
bond.ylds_exp <- bond.ylds_exp * 1200

{plot(us_gvt$Maturity, us_gvt$Mean,pch = 8, col = "blue", xlab = "Maturity in months", ylab = "Mean Yiel
lines(us_gvt$Maturity,bond.ylds_exp, type="l", col = "red")
legend("bottomright", legend = c("US T-sec yields", "Vasicek model mean yields"),
       pch = c(8, NA), col = c("blue", "red"), lty = c(0, 1))}
```




```r
## 5.2
# simulate the yield curve over the next four periods
innovations <- c(0.55, -0.28, 1.78, 0.19)
sigma <- sqrt((us_gvt$St.Dev/1200)^2*(1-varphi^2))
z.t1 <- varphi*bond.ylds_exp[1]/1200 + (1-varphi)*theta+sigma[1]*innovations[1]
z.t2 <- varphi*z.t1 + (1-varphi)*theta+sigma[1]*innovations[2]
z.t3 <- varphi*z.t2 + (1-varphi)*theta+sigma[1]*innovations[3]
z.t4 <- varphi*z.t3 + (1-varphi)*theta+sigma[1]*innovations[4]

B <- c()
B[1] <- 1
for (i in 2:120){
  B[i] <- 1+B[i-1]*varphi
}

A <- c()
A[1] <- 0
for (i in 2:120){
  A[i] <- A[i-1]+delta+B[i-1]*(1-varphi)*theta-((lambda+B[i-1]*sigma)^2)/2
}
```

```
A <- A[c(1,3,6,9,12,24,36,48,60,84,120)]
B <- B[c(1,3,6,9,12,24,36,48,60,84,120)]

curve1 <- A + B*z.t1
curve2 <- A + B*z.t2
curve3 <- A + B*z.t3
curve4 <- A + B*z.t4

curve1 <- curve1*us_gvt$Maturity^-1
curve2 <- curve2*us_gvt$Maturity^-1
curve3 <- curve3*us_gvt$Maturity^-1
curve4 <- curve4*us_gvt$Maturity^-1

curve1 <- curve1*1200
curve2 <- curve2*1200
curve3 <- curve3*1200
curve4 <- curve4*1200

{plot(us_gvt$Maturity, curve1, type = "l", ylim = c(5.2, 7.5), col = "blue", lwd = 2, xlab = "Maturity"
lines(us_gvt$Maturity, curve2, col = "red", lwd = 2)
lines(us_gvt$Maturity, curve3, col = "green", lwd = 2)
lines(us_gvt$Maturity, curve4, col = "purple", lwd = 2)
legend("bottomright", legend = c("t+1", "t+2", "t+3", "t+4"), col = c("blue", "red", "green", "purple")
```