FE515 2022A Final Exam

Yufu Liao

05/02/2023

Question 1: (50 points)

1.1

Download the historical prices for the ticker "SPY" from 2019-01-01 until now.

```
#install.packages('quantmod')
library(quantmod)
```

```
getSymbols(Symbols = "SPY", from = "2019-01-01", to = '2023-05-02')
```

```
## [1] "SPY"
```

```
SPY <- data.frame(SPY)
head(SPY)
```

	SPY.Open <dbl>	SPY.High <dbl>	SPY.Low <dbl>	SPY.Close <dbl>	SPY.Volume <dbl>	SPY.Adjusted <dbl>
2019-01-02	245.98	251.21	245.95	250.18	126925200	233.1717
2019-01-03	248.23	248.57	243.67	244.21	144140700	227.6076
2019-01-04	247.59	253.11	247.17	252.39	142628800	235.2315
2019-01-07	252.69	255.95	251.69	254.38	103139100	237.0862
2019-01-08	256.82	257.31	254.00	256.77	102512600	239.3137
2019-01-09	257.56	258.91	256.19	257.97	95006600	240.4321

6 rows

1.2

Calculate the daily log returns for SPY using the adjusted close prices.

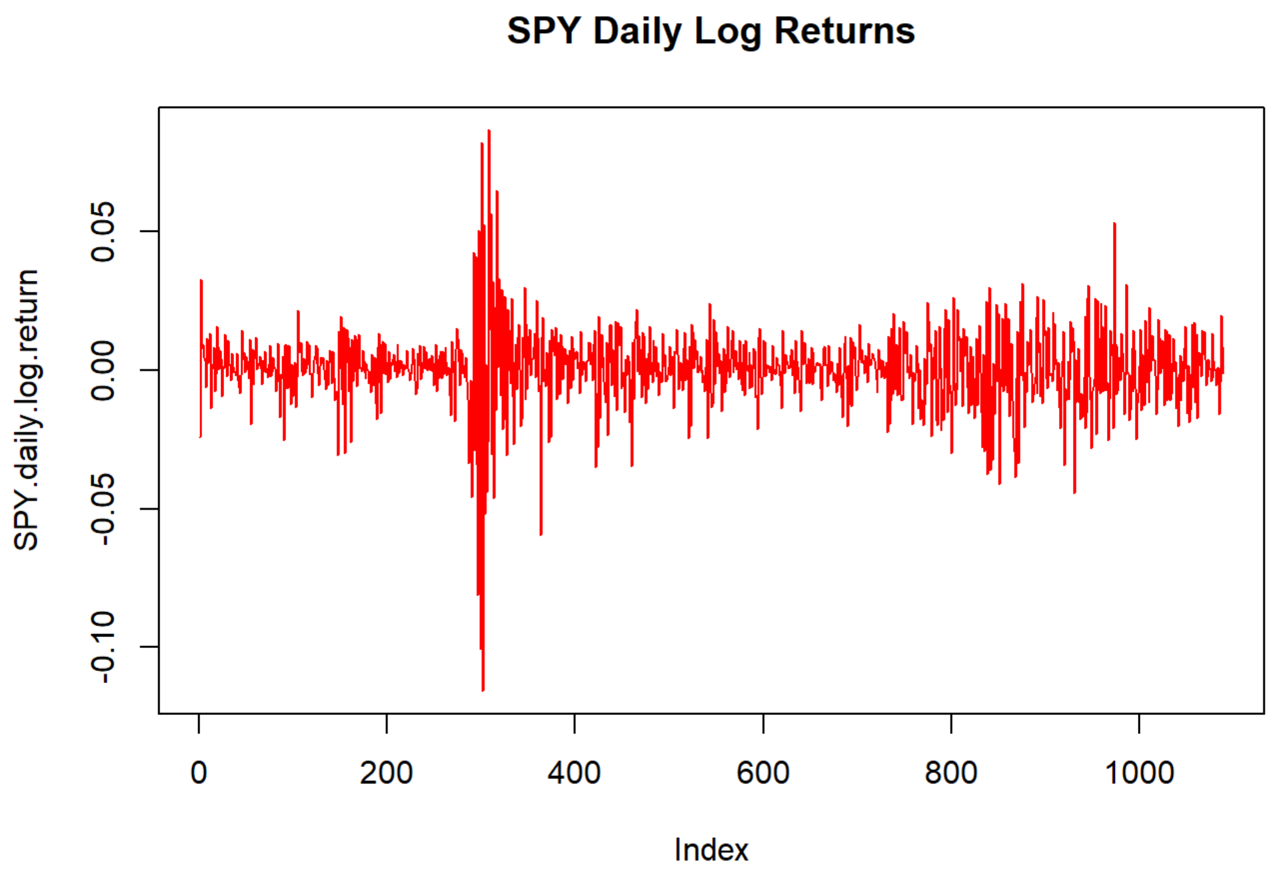
```
SPY.daily.log.return <- diff(log(SPY$SPY.Adjusted))

head(SPY.daily.log.return)
```

```
## [1] -0.024152074  0.032947049  0.007853549  0.009351429  0.004662830
## [6]  0.003521250
```

1.3

Plot the daily log returns in red line.

```
plot(SPY.daily.log.return, type = "l", col = "red", main = "SPY Daily Log Returns")
```



Question 2

2.1

Calculate the skewness and kurtosis of the SPY daily log return from Question 1, for both adjusted and unadjusted ones. (See page 21 and 23 of L6 and the corresponding HW problems)

```
library('moments')

SPY.daily.log.unadjusted_return <- diff(log(SPY$SPY.Close))

SPY.adjusted <- c(moments::skewness(SPY.daily.log.return), moments::kurtosis(SPY.daily.log.retur
n))
SPY.unadjusted <- c(moments::skewness(SPY.daily.log.unadjusted_return), moments::kurtosis(SPY.da
ily.log.unadjusted_return))

SPY.adjusted
```

```
## [1] -0.8044042 14.4146525
```

```
SPY.unadjusted
```

```
## [1] -0.8017831 14.2722111
```

2.2

Report the results in 2.1 using a 2×2 table (either data frame or matrix) such that: The column names are ”SPY.skewness” and ”SPY.kurtosis”. And the row names are ”Unadjusted” and ”Adjusted”.

```
table <- data.frame(
  "SPY.skewness" = c(SPY.adjusted[1], SPY.unadjusted[1]),
  "SPY.kurtosis" = c(SPY.adjusted[2], SPY.unadjusted[2]),
  row.names = c("Adjusted", "Unadjusted")
)
table
```

	SPY.skewness <dbl>	**SPY.kurtosis** <dbl>
Adjusted	-0.8044042	14.41465
Unadjusted	-0.8017831	14.27221

2 rows

Question 3

3.1

Download options prices for ticker ”SPY” for all expiration dates.

```
SPY.options.all <- getOptionChain("SPY", NULL)

length(SPY.options.all)
```

```
## [1] 31
```

3.2

For calls and puts of each expiration date, add a column of ”Price”, which is the average of “Bid” and “Ask”.

```
for(i in 1:length(SPY.options.all)){

  SPY.options.all[[i]]$calls$Price <-  rowMeans(SPY.options.all[[i]]$calls[, c('Bid', 'Ask')])
  SPY.options.all[[i]]$puts$Price <-  rowMeans(SPY.options.all[[i]]$puts[, c('Bid', 'Ask')])
}
```

3.3

For calls and puts of each expiration date, add a column of “ImpliedVol”, which is the implied volatility of the corresponding options calculated from root finding methods. (Method is not limited, but you may need to handle the problem when price difference has the same sign on the end of interval)

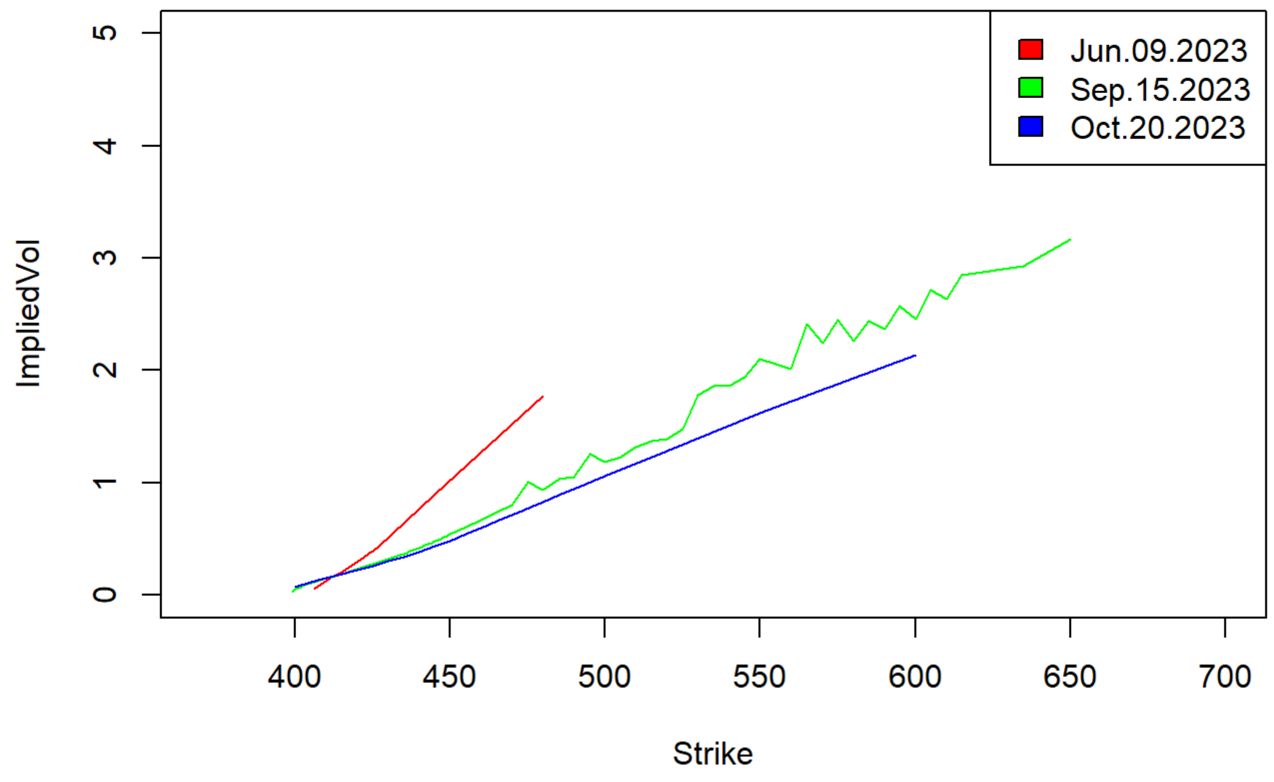
```r
bisection.new <- function(f, a, b, tol = 0.001, N.max = 100){
  # assign value
  f.a <- f(a)
  f.b <- f(b)

  # initial check
  if(is.na(f.a*f.b) || f.a*f.b > 0){# only modified this part
    return(NA)
  }else if(f.a == 0){
    return(a)
  }else if(f.b == 0){
    return(b)
  }

  # the same searching process
  for(n in 1:N.max){
    c <- (a+b)/2
    f.c <- f(c) # call function 'f' once each iteration
    if(f.c == 0 || abs(b - a) < tol){ # check absolute value
      break
    }
    if(f.a*f.c < 0){
      b <- c
      f.b <- f.c
    }else{
      a <- c
      f.a <- f.c
    }
  }

  return(c) # return a value rather than print it out
}

bs.call <- function(S0, K, T1, sigma, r){
  d1 <- (log(S0/K) + (r+0.5*sigma^2)*T1)/(sigma*sqrt(T1))
  d2 <- d1 - sigma*sqrt(T1)
  S0*pnorm(d1) - exp(-r*T1)*K*pnorm(d2)
  return(S0*pnorm(d1) - exp(-r*T1)*K*pnorm(d2))
}

# bs.call(100, 100, 1, r = 0.05, sigma = 0.2)


implied.vol.call <- function(S0, K, T1, r, price){
  price.diff <- function(sigma) bs.call(S0, K, T1, sigma, r) - price

  return(bisection.new(price.diff, 0.01, 5))

}

#implied.vol.call(S0,K,T1,r,10)
```

```
SPY.S0 <- getQuote("SPY")$Last
r <- 0.07 * 0.01
SPY.expiration <- names(SPY.options.all)
T.vec <- (as.Date(SPY.expiration,"%b.%d.%Y")-Sys.Date())/365
T.vec <- as.numeric(T.vec)

for(i in 1:length(SPY.options.all)){

  for(j in 1:nrow(SPY.options.all[[i]]$calls)){
    SPY.options.all[[i]]$calls$impliedVol[j] <-implied.vol.call(SPY.S0, SPY.options.all[[i]]$cal
ls$Strike[j], T.vec[i], r, SPY.options.all[[i]]$calls$Price[j])
  }
  SPY.options.all[[i]]$calls <- SPY.options.all[[i]]$calls[c("Bid", "Ask", "Strike","Price","imp
liedVol")]

  for(j in 1:nrow(SPY.options.all[[i]]$puts)){
    SPY.options.all[[i]]$puts$impliedVol[j] <-implied.vol.call(SPY.S0, SPY.options.all[[i]]$puts
$Strike[j], T.vec[i], r, SPY.options.all[[i]]$puts$Price[j])
  }
  SPY.options.all[[i]]$puts <- SPY.options.all[[i]]$puts[c("Bid", "Ask", "Strike","Price","impli
edVol")]
}
```

3.4

Choose 3 expiration date for put options, plot volatility smiles (Strike in x-axis and ImpliedVol in y-axis, similar to call smiles on page 22 of L9).

```
plot(NA, xlim = c(370,700), ylim = c(0,5), xlab = "Strike", ylab = "ImpliedVol")



lines(SPY.options.all[[14]]$puts$Strike, SPY.options.all[[14]]$puts$impliedVol,col = "red")
lines(SPY.options.all[[19]]$puts$Strike, SPY.options.all[[19]]$puts$impliedVol,col = "green")
lines(SPY.options.all[[21]]$puts$Strike, SPY.options.all[[21]]$puts$impliedVol,col = "blue")

SPY.expiration <- names(SPY.options.all)
legend("topright", SPY.expiration[c(14,19,21)], fill = c("red","green","blue"))
```



3.5

Keep fields “Strike”,“Bid”,“Ask”, “Price”, and “ImpliedVol” and save the calls and puts of each expiration date in .csv file. Submit one of the .csv file also.

```
for(i in 1:length(SPY.options.all)){

  write.csv(SPY.options.all[[i]]$calls, file = paste("SPY_opotion", i, "calls.csv", sep = ""))
  write.csv(SPY.options.all[[i]]$puts, file = paste("SPY_opotion", i, "puts.csv", sep = ""))

}
```