FE621 FinalFall2018

Shihao Zhang

2018-12-12

Problem A.

Pricing basket options

a&b.

```
#Given the information
A <- matrix(c(1,0.5,0.2,0.5,1,-0.4,0.2,-0.4,1),3,3)
A #correlation matrix
```

```
##      [,1] [,2] [,3]
## [1,]  1.0  0.5  0.2
## [2,]  0.5  1.0 -0.4
## [3,]  0.2 -0.4  1.0
```

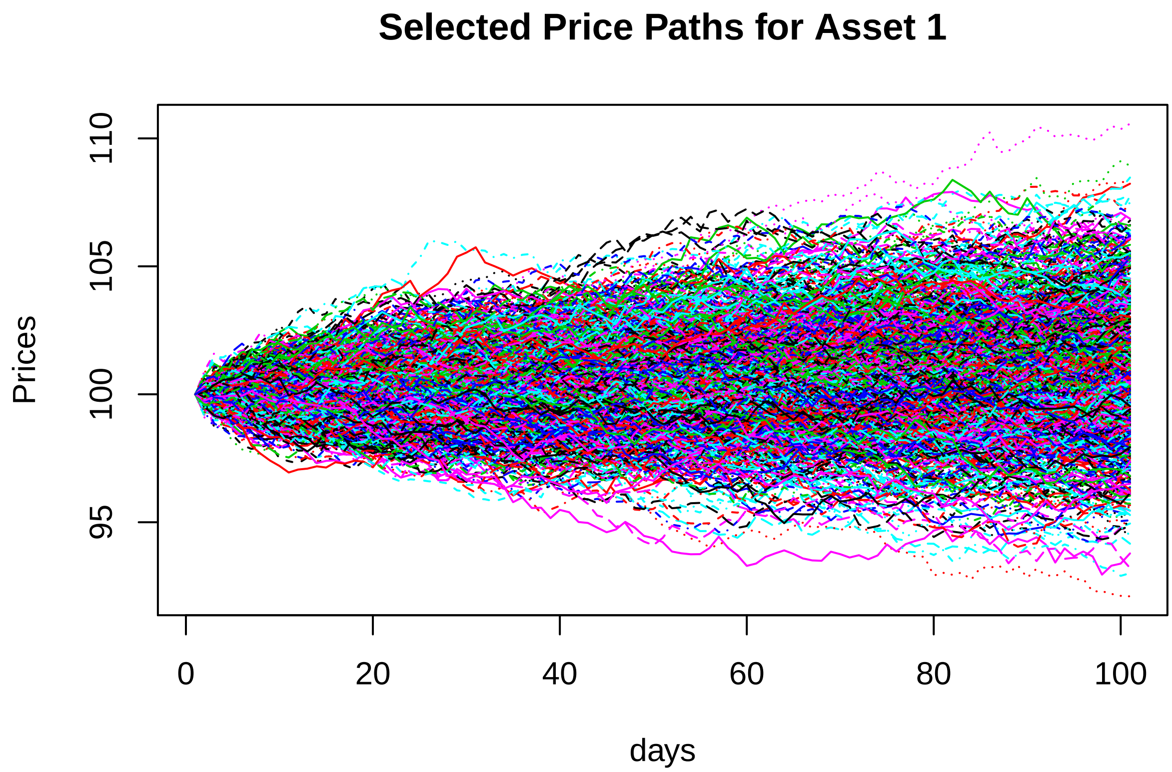
```
S0 <- c(100,101,98)
mu <- c(0.03,0.06,0.02)
sigma <- c(0.05,0.2,0.15)
n <- 1000 #trials(number of simulated paths)
m <- 100 #number of days
dt <- 1/365 #one day sampling frequency

#path simulation
path <- function(S0,mu,sigma,corr,dt,m,n){
  nassets <- length(S0)
  nu <- mu - sigma * sigma/2
  R <- chol(corr)
  S <- array(1, dim=c(m+1, n, nassets))
  for(i in 1:n)
  {
    x <- matrix(rnorm(m * nassets), ncol = nassets, nrow = m)
    ep <- x %*% R
    S[,i,] <- rbind(rep(1,nassets), apply(exp(matrix(nu*dt,nrow=m,ncol=nassets,byrow=TRUE) +
                                              (ep %*% diag(sigma)*sqrt(dt))), 2, function(x) cumprod(
  }
  return(S)
}
S <- path(S0,mu,sigma,A,dt,m,n)
S[,1:5,1] #an example of 5 trials of Price Paths for Asset 1
```

```
##            [,1]      [,2]      [,3]      [,4]      [,5]
##   [1,] 100.00000 100.00000 100.00000 100.00000 100.00000
##   [2,]  99.78640 100.31123 100.12196  99.91517  99.78970
##   [3,] 100.08522 100.13083 100.24673  99.86615  99.54673
##   [4,] 100.34076 100.24196 100.27647  99.97985  99.67402
##   [5,] 100.49085 100.49372 100.03719 100.23801 100.26200
##   [6,] 100.60125 100.60018  99.77406 100.38408 100.27656
```

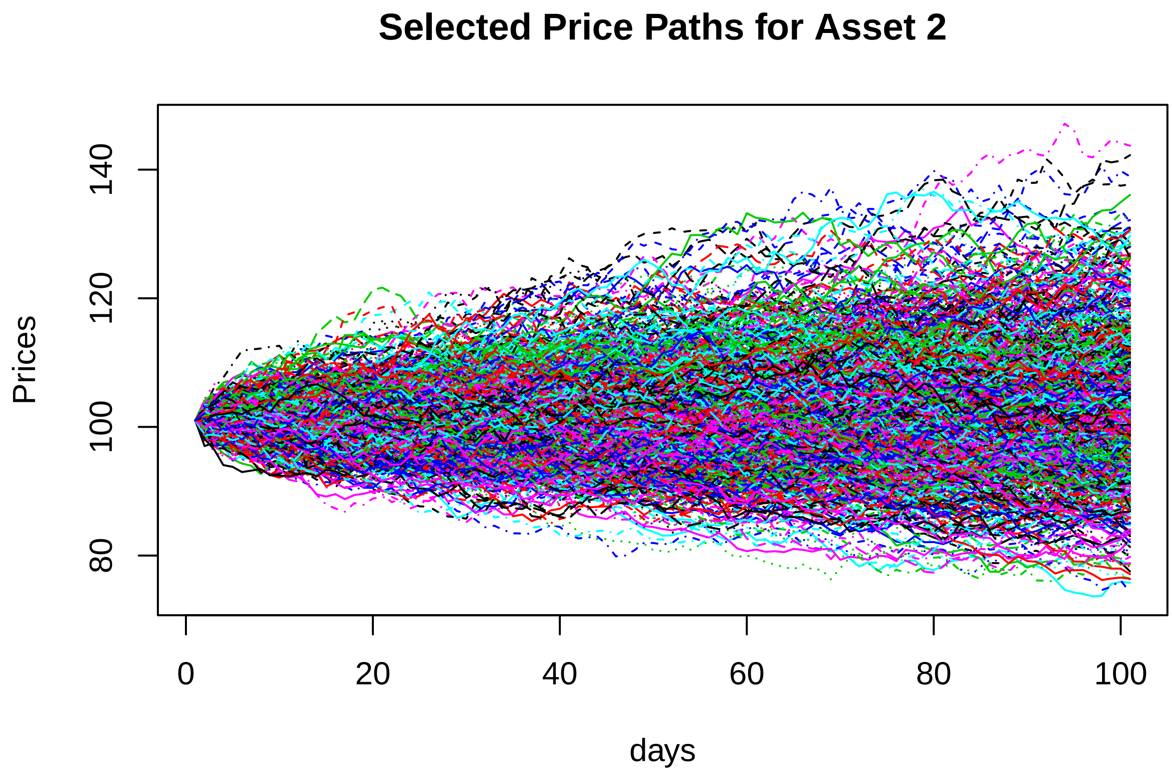
```
##   [7,] 101.14211 100.90528  99.60552 100.50563 100.34663
##   [8,] 100.71032 101.00478  99.60166 100.20816 100.39368
##   [9,] 100.19543 101.00081 100.23588 100.35444 100.04851
##  [10,]  99.80770 100.47278  99.93497 100.30610 100.20730
##  [11,]  99.84130  99.84005  99.68864  99.80093 100.25169
##  [12,]  99.88299  99.87564  99.79339  99.90401 100.30805
##  [13,]  99.85553  99.92166  99.60583  99.95263 100.22396
##  [14,]  99.83755 100.16571  99.35147  99.94865  99.94623
##  [15,] 100.00477 100.31331  99.68854  99.91879  99.95346
##  [16,]  99.77106 100.51633 100.25927  99.87423  99.59909
##  [17,]  99.44548 100.83062  99.93566  99.94477  99.64818
##  [18,]  99.66102 100.65239 100.23272 100.06693  99.17500
##  [19,]  99.34112 100.47449 100.01195 100.35456  99.14399
##  [20,]  99.46213 100.79372 100.15277 100.53837  99.07813
##  [21,]  99.51620 100.79855 100.37191 100.54909  99.38506
##  [22,]  99.70417 100.85167 100.14134 100.80454  99.51207
##  [23,]  99.81381 100.45482 100.43938 100.85471  99.57464
##  [24,]  99.68038 100.76219 100.22096 100.81754 100.04690
##  [25,]  99.45552 101.35286 100.58083 100.75750 100.11949
##  [26,]  99.88113 101.20948 100.54706 100.56623  99.91841
##  [27,]  99.83784 100.63458 100.63026 100.59380 100.10497
##  [28,]  99.61056 100.69657 100.79365 100.40274 100.24347
##  [29,]  99.33862 100.32394 101.07865 100.12212 100.83871
##  [30,]  99.29392 100.35774 101.32171  99.95780 100.90159
##  [31,]  98.66815 100.55242 101.84013 100.24510 100.89393
##  [32,]  98.48940 100.37525 101.34958  99.86919 100.50100
##  [33,]  98.45451 100.28129 101.24047 100.04239 100.97793
##  [34,]  98.39465 100.28023 101.41604 100.04125 100.91504
##  [35,]  98.41127  99.97777 101.26587  99.97211 100.61572
##  [36,]  97.85065  99.81083 101.52652  99.96334 100.26953
##  [37,]  97.88600  99.61732 102.09524  99.82522 100.20201
##  [38,]  97.89696  99.69954 101.63099 100.09409 100.19610
##  [39,]  98.04671  99.57536 101.96668 100.34512  99.94382
##  [40,]  97.71910  99.57909 101.32970  99.93429  99.70994
##  [41,]  97.13915  99.51575 101.47585  99.91991  99.70121
##  [42,]  97.59265  99.50616 102.01767 100.00294  99.43414
##  [43,]  97.01420  99.84043 102.25210 100.30368  99.71000
##  [44,]  96.71988  99.98924 102.24662 100.75608  99.52296
##  [45,]  96.63359 100.13602 102.35715 100.45278  99.91662
##  [46,]  96.76612 100.51517 102.39572 100.34617  99.77377
##  [47,]  96.87043 100.82158 102.34329 100.64082  99.97061
##  [48,]  97.31560 100.74905 102.46105 100.59135 100.29855
##  [49,]  97.54367 100.94635 102.31643 100.79448 100.42701
##  [50,]  97.58945 101.12853 101.97286 100.61821 100.16588
##  [51,]  97.55642 100.78442 101.90153 100.71773  99.98945
##  [52,]  97.31083 100.90943 102.12785 101.12534  99.89216
##  [53,]  97.65439 101.06730 102.12625 101.10880 100.03292
##  [54,]  97.61958 101.41223 102.41338 101.43097  99.80666
##  [55,]  97.76423 101.52435 101.97211 102.10967  99.60645
##  [56,]  97.60356 101.72776 102.57603 102.01478  99.44297
##  [57,]  97.49501 101.83764 102.21341 102.05477  99.54997
##  [58,]  97.14947 101.77544 101.86224 101.99615  99.22907
##  [59,]  96.99781 101.80649 101.90902 102.09144  99.11850
##  [60,]  97.09857 101.94846 101.84884 102.22219  99.21774
```

```
##  [61,]  96.83957 102.38254 102.32120 102.22000  99.70820
##  [62,]  96.99842 102.34587 102.32066 101.96268  99.41006
##  [63,]  96.89855 102.36256 102.56413 101.75611  99.30359
##  [64,]  96.80697 101.84588 102.35494 101.69088  98.71118
##  [65,]  96.56702 101.92097 102.67081 101.94878  98.72630
##  [66,]  96.64801 101.71770 102.95147 102.19204  99.01750
##  [67,]  96.43997 101.44730 103.04507 102.22991  99.13945
##  [68,]  95.97990 101.47232 103.27202 102.82868  99.22875
##  [69,]  96.10236 101.55472 103.03665 102.84778  99.45833
##  [70,]  96.39032 101.01344 103.38051 102.62072  99.08715
##  [71,]  96.01893 101.02672 103.43244 102.47002  99.23721
##  [72,]  95.86337 101.52313 103.41132 102.27498  99.22703
##  [73,]  96.26573 101.20582 103.54492 102.60750  99.33215
##  [74,]  96.37862 101.72711 103.79643 102.42731  98.98620
##  [75,]  96.09519 102.08869 104.10532 102.56891  98.97305
##  [76,]  96.20731 101.97500 103.77245 102.60528  98.64858
##  [77,]  96.57260 101.62529 103.68250 103.03455  98.59972
##  [78,]  96.63710 101.47472 103.43752 102.84638  99.07672
##  [79,]  96.75872 101.43908 103.55944 102.77615  98.82030
##  [80,]  97.00645 101.58997 103.27737 102.72528  98.74184
##  [81,]  97.17060 100.92437 102.91857 102.59791  98.90605
##  [82,]  97.86097 100.91740 102.55010 102.86190  99.11979
##  [83,]  97.58286 100.70843 102.24001 102.78434  99.12922
##  [84,]  97.15325 101.58822 102.37469 102.46691  99.00953
##  [85,]  97.14721 101.71567 102.36602 102.31115  98.94460
##  [86,]  97.49027 101.87495 102.48881 102.17769  99.11603
##  [87,]  97.64403 102.13158 102.39076 102.14093  99.32963
##  [88,]  97.31061 102.53768 102.12823 102.06259  98.85455
##  [89,]  96.62526 102.72796 101.51877 102.24273  98.91070
##  [90,]  96.67387 102.93588 101.75672 102.23913  98.70017
##  [91,]  96.47217 102.69864 101.86034 102.59552  98.39564
##  [92,]  96.56326 102.58989 101.53589 102.28096  97.97354
##  [93,]  96.53262 102.92177 101.86414 102.67307  98.45069
##  [94,]  96.69757 103.10437 101.66355 102.68471  98.54206
##  [95,]  97.36823 102.65857 101.88484 102.37761  98.37695
##  [96,]  97.49174 103.23352 101.96081 102.41809  98.01065
##  [97,]  97.25493 103.66547 102.13592 102.71595  98.34325
##  [98,]  97.36392 103.53906 102.19874 102.46746  98.50885
##  [99,]  97.07706 103.79778 102.35690 102.41854  98.70570
## [100,]  97.57324 103.77395 102.39866 102.12934  98.58356
## [101,]  97.51991 104.21840 102.66020 102.06870  98.52031
```

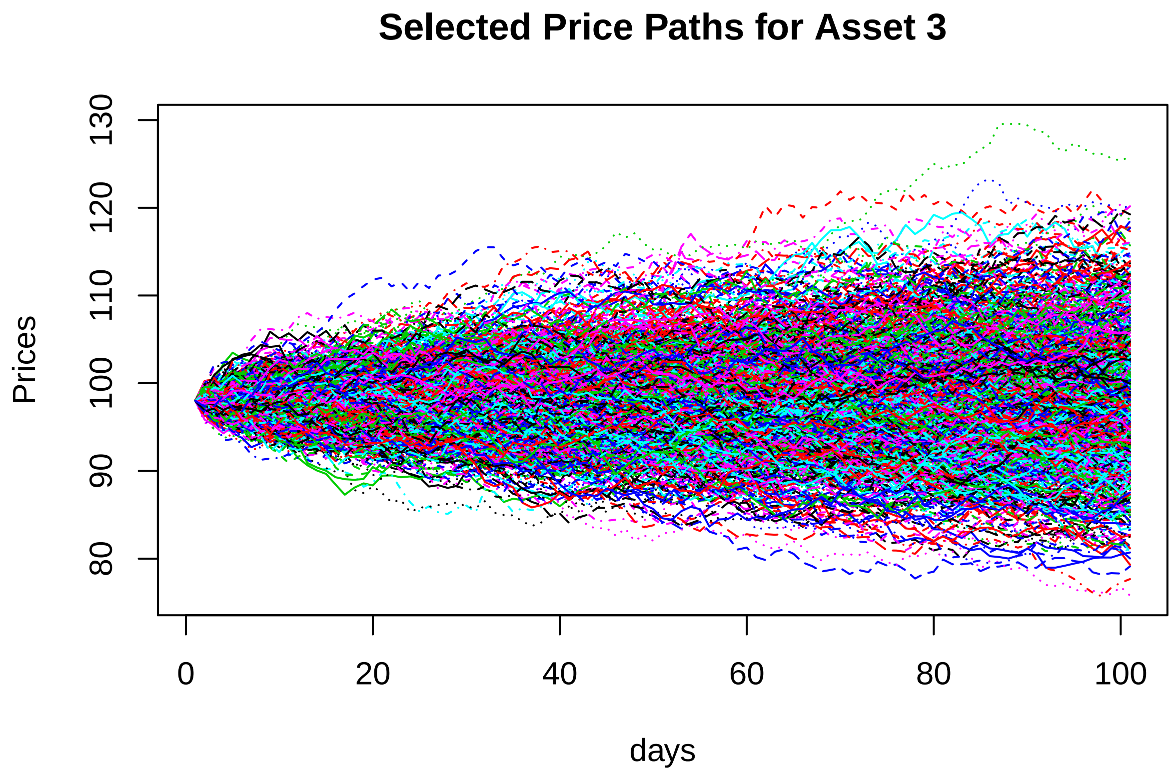
```
#Plot these 1000 sample paths
matplot(S[,1:1000,1],type='l', xlab='days', ylab='Prices',
        main='Selected Price Paths for Asset 1')
```




```
matplot(S[,1:1000,2],type='l', xlab='days', ylab='Prices',
        main='Selected Price Paths for Asset 2')
```




```r
matplot(S[,1:1000,3],type='l', xlab='days', ylab='Prices',
        main='Selected Price Paths for Asset 3')
```



c.Basket options

```
#Given the information
A <- matrix(c(1,0.5,0.2,0.5,1,-0.4,0.2,-0.4,1),3,3)
S0 <- c(100,101,98)
mu <- c(0.03,0.06,0.02)
sigma <- c(0.05,0.2,0.15)
n <- 10^6 #trials
m <- 100
dt <- 1/365
K <- 100
#Apply Monte Carlo simulation
Basket <- function(iscall,S0,mu,sigma,corr,dt,m,n){
  begintime<-Sys.time()
  if(iscall=="call"){cp <- 1} ##distinguish call and put option
  if(iscall=="put"){cp <- (-1)}

  nassets <- length(S0)
  nu <- mu - sigma * sigma/2
  R <- chol(corr)
  S <- array(1, dim=c(m+1, n, nassets))
  for(i in 1:n)
  {
    x <- matrix(rnorm(m * nassets), ncol = nassets, nrow = m)
    ep <- x %*% R
    S[,i,] <- rbind(rep(1,nassets), apply(exp(matrix(nu*dt,nrow=m,ncol=nassets,byrow=TRUE) +
                                                  (ep %*% diag(sigma)*sqrt(dt))), 2, function(x) cumprod(
```

```
  }
  #A vanilla basket option is simply a vanilla option on U(T)
  U <- c()
  for (i in 1:n) {
    U[i] <- max(0,cp*(S[(m+1),i,1]*(1/3)+S[(m+1),i,2]*(1/3)
                  +S[(m+1),i,3]*(1/3)-K))
  }
  U.avg <- mean(U)
  #Confidence interval
  upside.95 <- mean(U)+1.96*sd(U)/sqrt(n)
  downside.95 <- mean(U)-1.96*sd(U)/sqrt(n)

  endtime<-Sys.time()
  timecost<-endtime-begintime
  return(c(U.avg,upside.95,downside.95))
}

Basket.call <- Basket("call",S0,mu,sigma,A,dt,m,n)
Basket.put <- Basket("put",S0,mu,sigma,A,dt,m,n)
Basket.table <- cbind(Basket.call,Basket.put)
row.names(Basket.table) <- c("vanilla basket option value",
                             "95% Confidence interval upside",
                             "95% Confidence interval downside")
Basket.table #This result is obtained using 10^6 MC simulation
```

```
##                                  Basket.call Basket.put
## vanilla basket option value         1.998353   1.321600
## 95% Confidence interval upside      2.003682   1.325709
## 95% Confidence interval downside    1.993023   1.317491
```

d.Exotic options (i)

```
B <- 104 #barrier
#Condition is if the asset 2 hits the barrier
Basket.barrier1 <- function(iscall,B,S0,mu,sigma,corr,dt,m,n){
  begintime<-Sys.time()
  if(iscall=="call"){cp <- 1} ##distinguish call and put option
  if(iscall=="put"){cp <- (-1)}

  nassets <- length(S0)
  nu <- mu - sigma * sigma/2
  R <- chol(corr)
  S <- array(1, dim=c(m+1, n, nassets))
  for(i in 1:n)
  {
    x <- matrix(rnorm(m * nassets), ncol = nassets, nrow = m)
    ep <- x %*% R
    S[,i,] <- rbind(rep(1,nassets), apply(exp(matrix(nu*dt,nrow=m,ncol=nassets,byrow=TRUE) +
                                        (ep %*% diag(sigma)*sqrt(dt))), 2, function(x) cumprod(
  }

  max.asset2 <- apply(S[,,2],2,function(x)max(x))
  payoff <- c()
  for (i in 1:n) {
    #hits the barrier
```

```
    if(max.asset2[i]>B){
      payoff[i] <- max(0,cp*(S[(m+1),i,1]*(1/3)+S[(m+1),i,2]*(1/3)
                  +S[(m+1),i,3]*(1/3)-K))
    }
    #not hits the barrier
    if(max.asset2[i]<=B){
      payoff[i] <- 0
    }
  }

  payoff.avg <- mean(payoff)
  #Confidence interval
  upside.95 <- mean(payoff)+1.96*sd(payoff)/sqrt(n)
  downside.95 <- mean(payoff)-1.96*sd(payoff)/sqrt(n)

  endtime<-Sys.time()
  timecost<-endtime-begintime
  return(c(payoff.avg,upside.95,downside.95))
}
Basket.barrier.call1 <- Basket.barrier1("call",B,S0,mu,sigma,A,dt,m,n)
Basket.barrier.put1 <- Basket.barrier1("put",B,S0,mu,sigma,A,dt,m,n)
table41 <- cbind(Basket.barrier.call1,Basket.barrier.put1)
row.names(table41) <- c("basket barrier option value",
                             "95% Confidence interval upside",
                             "95% Confidence interval downside")
table41 #This result is obtained using 10^6 MC simulation
```

```
##                                  Basket.barrier.call1 Basket.barrier.put1
## basket barrier option value                  1.876918           0.6667509
## 95% Confidence interval upside               1.882260           0.6696985
## 95% Confidence interval downside             1.871576           0.6638032
```

(ii)

Professor, for this problem I assume that the payoff of the option is (S2-K)+, instead of (S2^2-K)+, because S2^2 makes no sense, no option is paid square of the stock price.

```
Basket.barrier2 <- function(iscall,B,S0,mu,sigma,corr,dt,m,n){
  begintime<-Sys.time()
  if(iscall=="call"){cp <- 1} ##distinguish call and put option
  if(iscall=="put"){cp <- (-1)}

  nassets <- length(S0)
  nu <- mu - sigma * sigma/2
  R <- chol(corr)
  S <- array(1, dim=c(m+1, n, nassets))
  for(i in 1:n)
  {
    x <- matrix(rnorm(m * nassets), ncol = nassets, nrow = m)
    ep <- x %*% R
    S[,i,] <- rbind(rep(1,nassets), apply(exp(matrix(nu*dt,nrow=m,ncol=nassets,byrow=TRUE) +
```

```
                                            (ep %*% diag(sigma)*sqrt(dt))), 2, function(x) cumprod(
  }

  max.asset2 <- apply(S[,,2],2,function(x)max(x))
  max.asset3 <- apply(S[,,3],2,function(x)max(x))
  payoff <- c()
  for (i in 1:n) {
    #Condition 1
    if(max.asset2[i]<=max.asset3[i]){
      payoff[i] <- max(0,cp*(S[(m+1),i,1]*(1/3)+S[(m+1),i,2]*(1/3)
                 +S[(m+1),i,3]*(1/3)-K))
    }
    #Condition 2
    if(max.asset2[i]>max.asset3[i]){
      payoff[i] <- max(0,cp*(S[(m+1),i,2]-K))
    }
  }

  payoff.avg <- mean(payoff)
  #Confidence interval
  upside.95 <- mean(payoff)+1.96*sd(payoff)/sqrt(n)
  downside.95 <- mean(payoff)-1.96*sd(payoff)/sqrt(n)

  endtime<-Sys.time()
  timecost<-endtime-begintime
  return(c(payoff.avg,upside.95,downside.95))
}
Basket.barrier.call2 <- Basket.barrier2("call",B,S0,mu,sigma,A,dt,m,n)
Basket.barrier.put2 <- Basket.barrier2("put",B,S0,mu,sigma,A,dt,m,n)
table42 <- cbind(Basket.barrier.call2,Basket.barrier.put2)
row.names(table42) <- c("basket barrier option value",
                        "95% Confidence interval upside",
                        "95% Confidence interval downside")
table42 #This result is obtained using 10^6 MC simulation
```

```
##                                  Basket.barrier.call2 Basket.barrier.put2
## basket barrier option value                  5.912841            1.513430
## 95% Confidence interval upside               5.927414            1.519506
## 95% Confidence interval downside             5.898269            1.507354
```

(iii)

```
Basket.barrier3 <- function(iscall,B,S0,mu,sigma,corr,dt,m,n){
  begintime<-Sys.time()
  if(iscall=="call"){cp <- 1} ##distinguish call and put option
  if(iscall=="put"){cp <- (-1)}

  nassets <- length(S0)
  nu <- mu - sigma * sigma/2
  R <- chol(corr)
  S <- array(1, dim=c(m+1, n, nassets))
  for(i in 1:n)
  {
    x <- matrix(rnorm(m * nassets), ncol = nassets, nrow = m)
    ep <- x %*% R
```

```
    S[,i,] <- rbind(rep(1,nassets), apply(exp(matrix(nu*dt,nrow=m,ncol=nassets,byrow=TRUE) +
                                            (ep %*% diag(sigma)*sqrt(dt))), 2, function(x) cumprod(
  }

  avg.asset2 <- apply(S[,,2],2,function(x)mean(x))
  avg.asset3 <- apply(S[,,3],2,function(x)mean(x))
  payoff <- c()
  for (i in 1:n) {
    #Condition 1
    if(avg.asset2[i]<=avg.asset3[i]){
      payoff[i] <- 0
    }
    #Condition 2
    if(avg.asset2[i]>avg.asset3[i]){
      payoff[i] <- max(0,cp*(avg.asset2[i]-K))
    }
  }

  payoff.avg <- mean(payoff)
  #Confidence interval
  upside.95 <- mean(payoff)+1.96*sd(payoff)/sqrt(n)
  downside.95 <- mean(payoff)-1.96*sd(payoff)/sqrt(n)

  endtime<-Sys.time()
  timecost<-endtime-begintime
  return(c(payoff.avg,upside.95,downside.95))
}
Basket.barrier.call3 <- Basket.barrier3("call",B,S0,mu,sigma,A,dt,m,n)
Basket.barrier.put3 <- Basket.barrier3("put",B,S0,mu,sigma,A,dt,m,n)
table43 <- cbind(Basket.barrier.call3,Basket.barrier.put3)
row.names(table43) <- c("basket barrier option value",
                             "95% Confidence interval upside",
                             "95% Confidence interval downside")
table43 #This result is obtained using 10^6 MC simulation
```

```
##                                  Basket.barrier.call3 Basket.barrier.put3
## basket barrier option value                  3.366639           0.1839852
## 95% Confidence interval upside               3.375244           0.1854131
## 95% Confidence interval downside             3.358034           0.1825573
```

Problem B.

Principal Component Analysis

1.Download daily prices,Construct the corresponding matrix of standardized returns.

```
library(quantmod)
```

```
## Loading required package: xts

## Loading required package: zoo

##
```

```
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Loading required package: TTR

## Version 0.4-0 included new data defaults. See ?getSymbols.
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'

## The following object is masked from 'package:base':
##
##     date
```

```
DJI <- get(getSymbols("^DJI", from="2013-12-13", to="2018-12-13"))
```

```
## 'getSymbols' currently uses auto.assign=TRUE by default, but will
## use auto.assign=FALSE in 0.5-0. You will still be able to use
## 'loadSymbols' to automatically load data. getOption("getSymbols.env")
## and getOption("getSymbols.auto.assign") will still be checked for
## alternate defaults.
##
## This message is shown once per session and may be disabled by setting
## options("getSymbols.warning4.0"=FALSE). See ?getSymbols for details.

##
## WARNING: There have been significant changes to Yahoo Finance data.
## Please see the Warning section of '?getSymbols.yahoo' for details.
##
## This message is shown once per session and may be disabled by setting
## options("getSymbols.yahoo.warning"=FALSE).
```

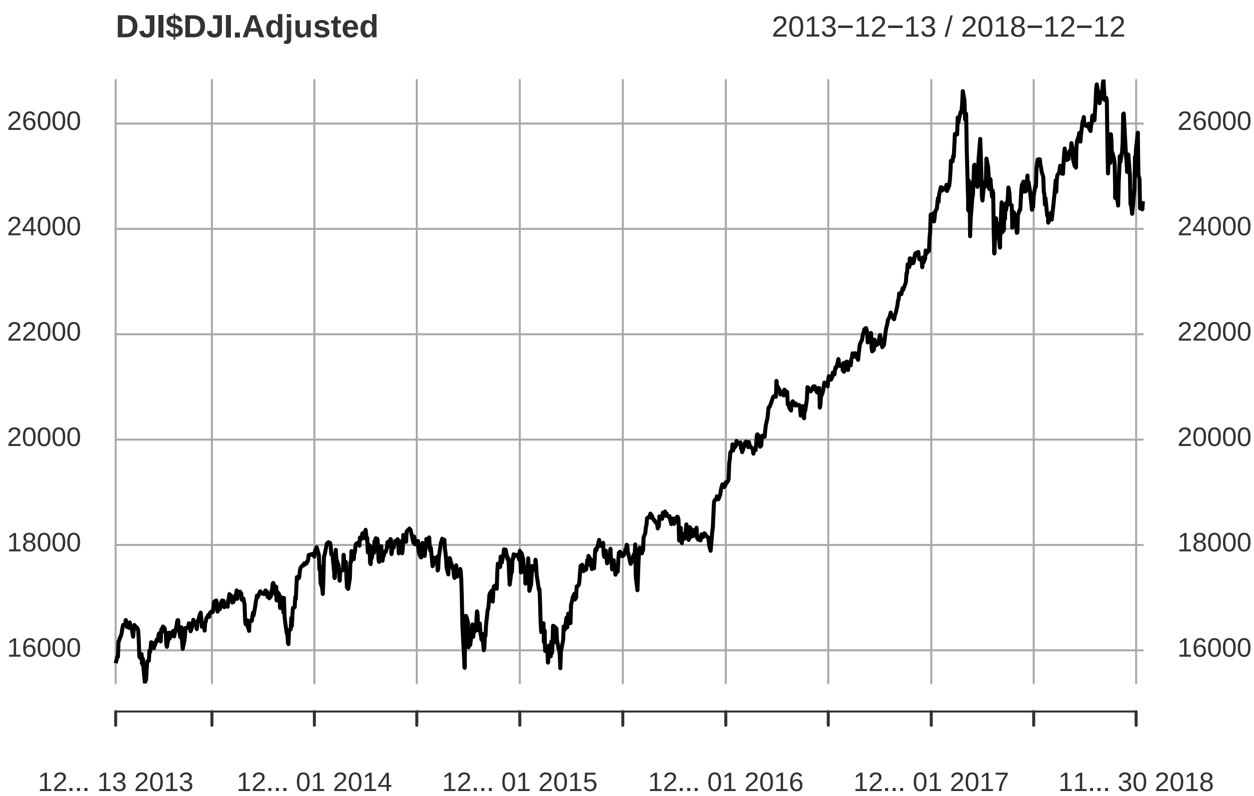
```
Date <- time(DJI) #Date vector
plot(DJI$DJI.Adjusted) #  Dow Jones Industrial Average for the last 5 years
```

```
## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 13 2013'  <e6>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 13 2013'  <9c>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 13 2013'  <88>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 13 2013'  <e6>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 13 2013'  <9c>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 13 2013'  <88>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '6 <U+FFFD> 02 2014'  <e6>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
```

```
## $las, : 'mbcsToSbcs'  '6 <U+FFFD> 02 2014'  <9c>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '6 <U+FFFD> 02 2014'  <88>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2014'  <e6>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2014'  <9c>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2014'  <88>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2014'  <e6>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2014'  <9c>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2014'  <88>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '6 <U+FFFD> 01 2015'  <e6>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '6 <U+FFFD> 01 2015'  <9c>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '6 <U+FFFD> 01 2015'  <88>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2015'  <e6>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2015'  <9c>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2015'  <88>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2015'  <e6>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2015'  <9c>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2015'  <88>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '6 <U+FFFD> 01 2016'  <e6>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '6 <U+FFFD> 01 2016'  <9c>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '6 <U+FFFD> 01 2016'  <88>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2016'  <e6>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2016'  <9c>  dot
```

```
## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2016' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2016' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2016' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2016' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 01 2017' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 01 2017' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 01 2017' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2017' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2017' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2017' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2017' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2017' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2017' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 01 2018' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 01 2018' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 01 2018' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '11 <U+FFFD> 30 2018' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '11 <U+FFFD> 30 2018' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '11 <U+FFFD> 30 2018' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '11 <U+FFFD> 30 2018' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '11 <U+FFFD> 30 2018' <9c> dot
```

```
## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '11 <U+FFFD> 30 2018' <88> dot
```




```
#Download components of DJIA stock price
ticker <- c("WMT","DIS","CAT","XOM","IBM",
            "UNH","HD","INTC","AXP","MRK",
            "UTX","MMM","CVX","CSCO","AAPL",
            "MCD","KO","V","WBA","JNJ",
            "PFE","MSFT","PG","JPM","VZ",
            "DWDP","GS","BA","NKE","TRV")
#1~5
WMT.P <- get(getSymbols("WMT", from="2013-12-13", to="2018-12-13"))
WMT <- WMT.P$WMT.Close
DIS.P <- get(getSymbols("DIS", from="2013-12-13", to="2018-12-13"))
DIS <- DIS.P$DIS.Close
CAT.P <- get(getSymbols("CAT", from="2013-12-13", to="2018-12-13"))
CAT <-CAT.P$CAT.Close
XOM.P <- get(getSymbols("XOM", from="2013-12-13", to="2018-12-13"))
XOM <-XOM.P$XOM.Close
IBM.P <- get(getSymbols("IBM", from="2013-12-13", to="2018-12-13"))
IBM <-IBM.P$IBM.Close
#6~10
UNH.P <- get(getSymbols("UNH", from="2013-12-13", to="2018-12-13"))
UNH <- UNH.P$UNH.Close
HD.P <- get(getSymbols("HD", from="2013-12-13", to="2018-12-13"))
HD <- HD.P$HD.Close
INTC.P <- get(getSymbols("INTC", from="2013-12-13", to="2018-12-13"))
```

```
INTC <- INTC.P$INTC.Close
AXP.P <- get(getSymbols("AXP", from="2013-12-13", to="2018-12-13"))
AXP <- AXP.P$AXP.Close
MRK.P <- get(getSymbols("MRK", from="2013-12-13", to="2018-12-13"))
MRK <- MRK.P$MRK.Close
#11~15
UTX.P <- get(getSymbols("UTX", from="2013-12-13", to="2018-12-13"))
UTX <- UTX.P$UTX.Close
MMM.P <- get(getSymbols("MMM", from="2013-12-13", to="2018-12-13"))
MMM <- MMM.P$MMM.Close
CVX.P <- get(getSymbols("CVX", from="2013-12-13", to="2018-12-13"))
CVX <- CVX.P$CVX.Close
CSCO.P <- get(getSymbols("CSCO", from="2013-12-13", to="2018-12-13"))
CSCO <- CSCO.P$CSCO.Close
AAPL.P <- get(getSymbols("AAPL", from="2013-12-13", to="2018-12-13"))
AAPL <- AAPL.P$AAPL.Close
#16~20
MCD.P <- get(getSymbols("MCD", from="2013-12-13", to="2018-12-13"))
MCD <- MCD.P$MCD.Close
KO.P <- get(getSymbols("KO", from="2013-12-13", to="2018-12-13"))
KO <- KO.P$KO.Close
V.P <- get(getSymbols("V", from="2013-12-13", to="2018-12-13"))
V <- V.P$V.Close
WBA.P <- get(getSymbols("WBA", from="2013-12-13", to="2018-12-13"))
WBA <- WBA.P$WBA.Close
JNJ.P <- get(getSymbols("JNJ", from="2013-12-13", to="2018-12-13"))
JNJ <- JNJ.P$JNJ.Close
#21~25
PFE.P <- get(getSymbols("PFE", from="2013-12-13", to="2018-12-13"))
PFE <- PFE.P$PFE.Close
MSFT.P <- get(getSymbols("MSFT", from="2013-12-13", to="2018-12-13"))
MSFT <- MSFT.P$MSFT.Close
PG.P <- get(getSymbols("PG", from="2013-12-13", to="2018-12-13"))
PG <- PG.P$PG.Close
JPM.P <- get(getSymbols("JPM", from="2013-12-13", to="2018-12-13"))
JPM <- JPM.P$JPM.Close
VZ.P <- get(getSymbols("VZ", from="2013-12-13", to="2018-12-13"))
VZ <- VZ.P$VZ.Close
#26~30
DWDP.P <- get(getSymbols("DWDP", from="2013-12-13", to="2018-12-13"))
DWDP <- DWDP.P$DWDP.Close
GS.P <- get(getSymbols("GS", from="2013-12-13", to="2018-12-13"))
GS <- GS.P$GS.Close
BA.P <- get(getSymbols("BA", from="2013-12-13", to="2018-12-13"))
BA <- BA.P$BA.Close
NKE.P <- get(getSymbols("NKE", from="2013-12-13", to="2018-12-13"))
NKE <- NKE.P$NKE.Close
TRV.P <- get(getSymbols("TRV", from="2013-12-13", to="2018-12-13"))
TRV <- TRV.P$TRV.Close

#Now combine the data
#components of DJIA
DJIA <- cbind(WMT,DIS,CAT,XOM,IBM,
```

```
            UNH,HD,INTC,AXP,MRK,
            UTX,MMM,CVX,CSCO,AAPL,
            MCD,KO,V,WBA,JNJ,
            PFE,MSFT,PG,JPM,VZ,
            DWDP,GS,BA,NKE,TRV)
T <- nrow(DJIA)
T #1258 days
```

```
## [1] 1258
```

```
N <- ncol(DJIA)
N #30 components stock
```

```
## [1] 30
```

```
DJIA <- as.data.frame(DJIA)
#Construct matrix of standardized returns
#daily log return R
R <- matrix(NA,nrow = nrow(DJIA),ncol = ncol(DJIA))
R[1,] <- 0
for (j in 1:ncol(DJIA)) {
  for (i in 2:nrow(DJIA)) {
    R[i,j] <- log(DJIA[i,j]/DJIA[i-1,j])
  }
}
#daily mean return&std
options(scipen = 200,digits=6) #do not use Scientific notation
Rmean <- apply(R,2,mean)
Std <- c()
for (i in 1:30) {
  Std[i] <- sqrt(sum((R[,i]-Rmean[i])^2)/T)
}
#matrix of standardized returns
Y <- matrix(NA,nrow = nrow(DJIA),ncol = ncol(DJIA))
for (j in 1:N) {
  for (i in 1:T) {
    Y[i,j] <- (R[i,j]-Rmean[j])/Std[j]
  }
}
colnames(Y) <- ticker
head(Y)
```

```
##             WMT        DIS        CAT        XOM        IBM        UNH
## [1,] -0.0115821 -0.0322404 -0.0187016  0.0153761  0.0224782 -0.0842667
## [2,] -0.3727679  1.0471182  0.9401283  1.7125756  2.3168464  0.5614515
## [3,] -0.5348947  0.1483340 -0.3486668 -0.3991463 -0.9190740 -0.4510065
## [4,]  0.7243853  1.7997638  0.8739123  2.4471233  1.3437911  1.7326171
## [5,] -0.7582661  0.8691683 -0.4669837 -0.0792023  0.6971071 -0.0733798
## [6,]  0.1917572 -0.6985915  0.9661259 -0.6322732 -0.0659615  0.9537249
##              HD       INTC        AXP        MRK         UTX        MMM
## [1,] -0.0546049 -0.0350379 -0.0154909 -0.0309225 -0.00784228 -0.0333603
## [2,]  0.0442963  0.3920510  0.7621414 -0.5204683  0.98563487  0.8544135
## [3,] -0.5834016  0.5212917 -0.3940769 -0.3532582 -0.77303423  2.6074343
## [4,]  1.5113903  1.2448613  1.7055431  1.6514603  1.82287773  2.9938192
## [5,] -0.1632083 -0.0609107  0.3638573 -0.2142939 -0.26475425  0.3776069
```

```
## [6,]  0.0431428 -0.2423711  1.0227126  0.8488102  0.94981181  0.1747858
##              CVX       CSCO       AAPL        MCD         KO          V
## [1,]  0.00211001 -0.0516341 -0.0405496 -0.0513194 -0.0208732 -0.0599482
## [2,]  0.19674578  1.5891681  0.3308400  0.9827590  0.0970870  0.0849987
## [3,] -0.90246598  0.8286952 -0.3440403 -1.1471813 -0.5230423  1.9556910
## [4,]  1.74016135  0.2395661 -0.5539061  1.5321614  2.6710778  0.6959786
## [5,]  0.96855554  0.2022575 -0.8155334 -0.8551599 -0.4845601  0.2010969
## [6,] -0.25911943  0.1653132  0.5203954  1.3384807  0.5006475 -0.0992278
##             WBA        JNJ        PFE       MSFT         PG        JPM
## [1,] -0.0187765 -0.0398339 -0.0274648 -0.0605699 -0.0114665 -0.0357284
## [2,] -0.1649711 -0.0168019 -0.0274648  0.3196024 -0.9146730  0.3043465
## [3,] -0.7320422 -0.8603651 -0.3601674 -0.7655195 -1.0567980 -0.9917659
## [4,]  1.2103119  2.2326285  1.8617976  0.0542343  1.9502903  2.0252766
## [5,] -0.4670841 -0.7918767 -0.2057228 -0.6943203 -0.6481609 -0.0491106
## [6,]  2.3005798  0.0858734 -1.4057666  0.9925044 -0.0913244  0.5905991
##              VZ       DWDP         GS         BA         NKE        TRV
## [1,] -0.0133847 -0.0141349 -0.00271046 -0.0489264 -0.03687007 -0.0268845
## [2,]  0.8062539  0.1129779  1.06532078  0.4082037 -0.00039212  0.6075022
## [3,] -1.3834571  0.9315190 -0.19343531  0.5423740  0.25426817 -0.5703919
## [4,]  1.7638437  2.0046405  1.78068708 -0.2471598  1.56962990  1.7397777
## [5,] -0.0908028  0.5751512 -0.03105280 -0.2120096 -0.29464170  0.6953453
## [6,] -0.6935131 -0.1948969  0.15507055  0.7121988 -0.86099593  0.4039174
```

2.Calculate the sample correlation matrix

```r
C <- cor(Y)
head(C) #sample correlation matrix
```

```
##          WMT      DIS      CAT      XOM      IBM      UNH       HD
## WMT 1.000000 0.276043 0.210356 0.240805 0.260441 0.296711 0.369994
## DIS 0.276043 1.000000 0.353744 0.393883 0.378936 0.367291 0.415887
## CAT 0.210356 0.353744 1.000000 0.523152 0.410525 0.331588 0.392595
## XOM 0.240805 0.393883 0.523152 1.000000 0.404126 0.341890 0.347283
## IBM 0.260441 0.378936 0.410525 0.404126 1.000000 0.333069 0.394939
## UNH 0.296711 0.367291 0.331588 0.341890 0.333069 1.000000 0.418590
##         INTC      AXP      MRK      UTX      MMM      CVX     CSCO
## WMT 0.231140 0.231781 0.296854 0.309936 0.307600 0.199166 0.327361
## DIS 0.389956 0.388719 0.311364 0.409325 0.430770 0.342229 0.454706
## CAT 0.433406 0.428662 0.294590 0.517524 0.551033 0.526538 0.454130
## XOM 0.379430 0.322200 0.358005 0.406980 0.477153 0.765940 0.422999
## IBM 0.415212 0.384509 0.353900 0.459966 0.442721 0.372864 0.467124
## UNH 0.347283 0.420223 0.365778 0.401458 0.425783 0.316936 0.360623
##         AAPL      MCD       KO        V      WBA      JNJ      PFE
## WMT 0.227530 0.290538 0.332478 0.256987 0.297406 0.339324 0.307750
## DIS 0.349704 0.288166 0.307278 0.439258 0.339909 0.348644 0.372375
## CAT 0.378053 0.270786 0.242233 0.424561 0.250549 0.327613 0.310204
## XOM 0.306575 0.305218 0.329610 0.377263 0.245560 0.420843 0.363421
## IBM 0.335048 0.286537 0.322378 0.430511 0.248611 0.371252 0.351929
## UNH 0.358756 0.307788 0.277625 0.431528 0.350980 0.418826 0.463414
##         MSFT       PG      JPM       VZ     DWDP       GS       BA
## WMT 0.259207 0.356673 0.266162 0.306212 0.205927 0.231280 0.278808
## DIS 0.384435 0.313843 0.473112 0.320135 0.404939 0.476270 0.409485
## CAT 0.437947 0.234095 0.532139 0.217684 0.547903 0.546784 0.511329
## XOM 0.363595 0.351896 0.499810 0.324885 0.500591 0.472164 0.404264
## IBM 0.450592 0.316082 0.450553 0.325395 0.396924 0.425273 0.401108
```

```
## UNH 0.410058 0.276203 0.450474 0.245511 0.369901 0.433083 0.397444
##          NKE      TRV
## WMT 0.270871 0.304204
## DIS 0.401932 0.408978
## CAT 0.308205 0.391753
## XOM 0.264981 0.412876
## IBM 0.297322 0.394103
## UNH 0.343479 0.418160
```

3.Calculate the eigenvalues and eigenvectors

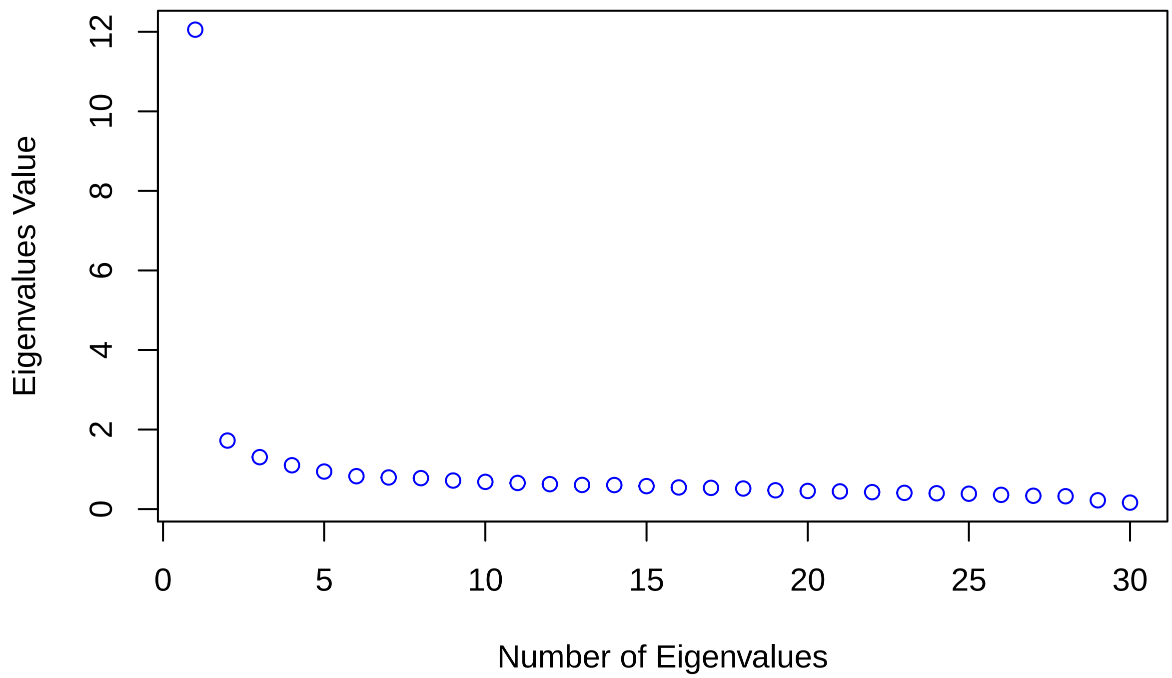
```r
decomposition <- eigen(C)
eigenvalues <- decomposition$values
eigenvectors <- decomposition$vectors
head(eigenvalues)
```

```
## [1] 12.053132  1.723258  1.306118  1.102268  0.943730  0.827491
```

```r
head(eigenvectors)
```

```
##           [,1]       [,2]       [,3]        [,4]       [,5]       [,6]
## [1,] -0.133471  0.2899172 -0.1326181 -0.05568349  0.1195969 -0.3685770
## [2,] -0.182174 -0.0272448 -0.0925347 -0.00330296  0.1221694 -0.1388923
## [3,] -0.191317 -0.2773719  0.2063094 -0.07090240  0.0462543 -0.0433383
## [4,] -0.188848 -0.0789100  0.4736885 -0.04380460 -0.0873024 -0.2963351
## [5,] -0.181580 -0.0364159  0.0299855 -0.12786094 -0.1206848  0.2118760
## [6,] -0.177738  0.0184814 -0.1748226  0.23178112 -0.0504860 -0.1126588
##            [,7]       [,8]       [,9]      [,10]      [,11]       [,12]
## [1,]  0.3890111  0.1598883 -0.1002918  0.6219102  0.1021210  0.03032923
## [2,]  0.0926829 -0.3726023  0.1779038 -0.1562578  0.1826204 -0.49027238
## [3,]  0.1138805  0.2108290 -0.0609916 -0.0231962 -0.1200111  0.07193870
## [4,] -0.1219644 -0.0509940  0.0567045  0.0589156  0.1299651  0.00993364
## [5,]  0.2479352 -0.0840993  0.2381435  0.1461336 -0.0367065  0.33042510
## [6,] -0.1621060  0.1919361 -0.1355709  0.0170935  0.2065465 -0.12463400
##           [,13]      [,14]      [,15]      [,16]      [,17]      [,18]
## [1,] -0.0847148  0.1382252 -0.0138987 -0.1528514  0.0209803  0.0416020
## [2,]  0.1845200 -0.0123366  0.3871181 -0.3610620  0.0119027 -0.0529162
## [3,] -0.0420189  0.1496865 -0.1278875  0.0126501  0.0904563  0.0980263
## [4,] -0.0794790 -0.0559894  0.1104281  0.0676283 -0.1387379 -0.0396631
## [5,]  0.2638128 -0.3635459  0.4180955  0.2659929  0.3018598  0.1081589
## [6,] -0.2058555 -0.6794112 -0.0776827 -0.0827660 -0.0292693  0.2688573
##           [,19]      [,20]       [,21]       [,22]      [,23]      [,24]
## [1,] -0.1531749  0.0695375 -0.13457576 -0.02139329  0.0342564  0.1979744
## [2,] -0.0267506 -0.0113316  0.15432400 -0.05357714  0.2403134  0.0763904
## [3,] -0.0999438 -0.0496665  0.26489164 -0.02143318  0.4868472 -0.1558998
## [4,]  0.0139736 -0.0255525  0.01280610 -0.07043383 -0.0408655  0.0945558
## [5,] -0.0149256 -0.2005256 -0.11086794 -0.00315215  0.0200718  0.0901632
## [6,] -0.0800901 -0.0826943  0.00508223  0.23052345  0.0565247 -0.1558841
##            [,25]       [,26]      [,27]       [,28]      [,29]       [,30]
## [1,] -0.03567981 -0.00659016 -0.0147010 -0.00629251  0.0323355 -0.01891499
## [2,]  0.00893027 -0.12503997  0.0116052 -0.13328460  0.0572155  0.02649698
## [3,] -0.33453922 -0.22754951  0.4103852  0.11811589 -0.0266427  0.06278880
## [4,]  0.08334435  0.05603625 -0.1170248  0.08335655 -0.7074326  0.02017337
## [5,] -0.11473258 -0.01223761 -0.0318248  0.06600422  0.0378833 -0.01836023
## [6,]  0.11211857 -0.14742722  0.0456379  0.08960702  0.0148229  0.00177042
```

```
#graph the eigenvalues
plot(1:length(eigenvalues),eigenvalues, col="blue",
     xlab="Number of Eigenvalues", ylab="Eigenvalues Value")
```




```
#What percent of the trace is explained by summing the first
#5 eigenvalues
percentoftrace <- sum(eigenvalues[1:5])/sum(eigenvalues)
percentoftrace
```

```
## [1] 0.57095
```

```
#This means 57.095% of the trace is explained the first 5 eigenvalues.
```

4.Calculate the sample mean and sample standard deviation of the factor F

```
#use the first eigenvalue as lambda1 and the first 30*1 eigenvector
Ft <- as.matrix(R)%*%(eigenvectors[,1]/Std)/sqrt(eigenvalues[1])
mean(Ft)
```

```
## [1] -0.04467
```

```
sd(Ft)
```

```
## [1] 1.0004
```

5.Why F and the particular market index might be related

```
#Download DIA
DIA.P <- get(getSymbols("DIA", from="2013-12-13", to="2018-12-13"))
plot(DIA.P$DIA.Close) #Dow Jones Industrial Average ETF
```

```
## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 13 2013' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 13 2013' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 13 2013' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 13 2013' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 13 2013' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 13 2013' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 02 2014' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 02 2014' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 02 2014' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2014' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2014' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2014' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2014' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2014' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2014' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 01 2015' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 01 2015' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 01 2015' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2015' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2015' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2015' <88> dot
```

```
## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2015'  <e6>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2015'  <9c>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2015'  <88>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '6 <U+FFFD> 01 2016'  <e6>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '6 <U+FFFD> 01 2016'  <9c>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '6 <U+FFFD> 01 2016'  <88>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2016'  <e6>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2016'  <9c>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2016'  <88>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2016'  <e6>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2016'  <9c>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2016'  <88>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '6 <U+FFFD> 01 2017'  <e6>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '6 <U+FFFD> 01 2017'  <9c>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '6 <U+FFFD> 01 2017'  <88>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2017'  <e6>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2017'  <9c>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2017'  <88>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2017'  <e6>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2017'  <9c>  dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs'  '12 <U+FFFD> 01 2017'  <88>  dot
```

```
## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 01 2018' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 01 2018' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 01 2018' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '11 <U+FFFD> 30 2018' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '11 <U+FFFD> 30 2018' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '11 <U+FFFD> 30 2018' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '11 <U+FFFD> 30 2018' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '11 <U+FFFD> 30 2018' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '11 <U+FFFD> 30 2018' <88> dot
```

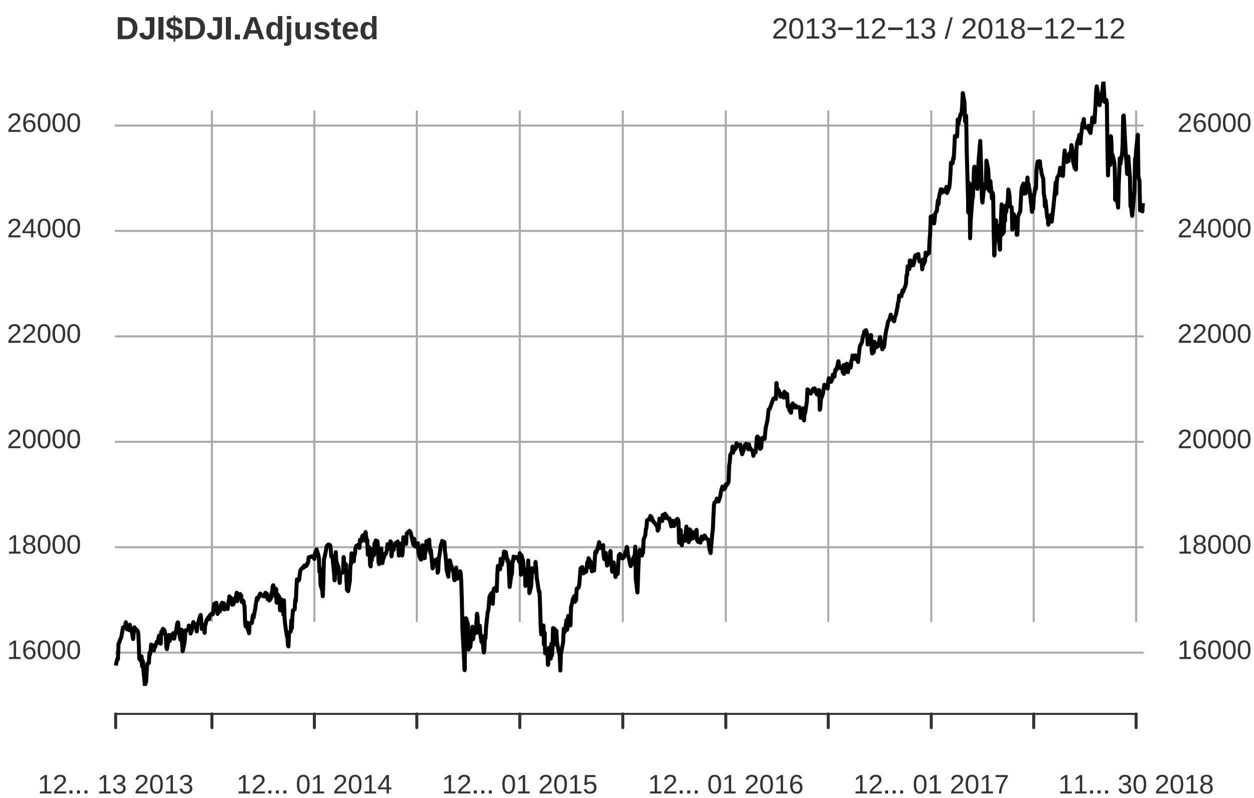

```
plot(DJI$DJI.Adjusted) #  Dow Jones Industrial Average for the last 5 years
```

```
## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 13 2013' <e6> dot
```

```
## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 13 2013' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 13 2013' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 13 2013' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 13 2013' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 13 2013' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 02 2014' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 02 2014' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 02 2014' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2014' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2014' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2014' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2014' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2014' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2014' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 01 2015' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 01 2015' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 01 2015' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2015' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2015' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2015' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2015' <e6> dot
```

```
## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2015' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2015' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 01 2016' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 01 2016' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 01 2016' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2016' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2016' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2016' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2016' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2016' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2016' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 01 2017' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 01 2017' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 01 2017' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2017' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2017' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2017' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2017' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2017' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '12 <U+FFFD> 01 2017' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 01 2018' <e6> dot
```

```
## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 01 2018' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '6 <U+FFFD> 01 2018' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '11 <U+FFFD> 30 2018' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '11 <U+FFFD> 30 2018' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '11 <U+FFFD> 30 2018' <88> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '11 <U+FFFD> 30 2018' <e6> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '11 <U+FFFD> 30 2018' <9c> dot

## Warning in axis(1, at = xycoords$x[axt], labels = names(axt), las = theme
## $las, : 'mbcsToSbcs' '11 <U+FFFD> 30 2018' <88> dot
```




```r
DIA <- DIA.P$DIA.Close
DIA <- as.matrix(DIA)
#Calculate standardized return for DIA
#daily log return R.DIA
R.DIA <- matrix(NA,nrow = nrow(DIA),ncol=1)
R.DIA[1,1] <- 0
```

```
for (j in 1:ncol(DIA)) {
  for (i in 2:nrow(DIA)) {
    R.DIA[i,j] <- log(DIA[i,j]/DIA[i-1,j])
  }
}
#daily mean return&std of DIA
R.DIAmean <- apply(R.DIA,2,mean)
Std.DIA <- sqrt(sum((R.DIA-R.DIAmean)^2)/T)
#standardized returns of DIA
sdreturn.DIA <- (R.DIA-R.DIAmean)/Std.DIA
#Linear Regression
lm <- lm(sdreturn.DIA~Ft)
summary(lm)
```

```
##
## Call:
## lm(formula = sdreturn.DIA ~ Ft)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.7408 -0.0857  0.0091  0.0914  0.6151
##
## Coefficients:
##             Estimate Std. Error t value            Pr(>|t|)
## (Intercept) -0.04412    0.00442   -9.98 <0.0000000000000002 ***
## Ft          -0.98767    0.00442 -223.60 <0.0000000000000002 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.157 on 1256 degrees of freedom
## Multiple R-squared:  0.975,  Adjusted R-squared:  0.975
## F-statistic: 5e+04 on 1 and 1256 DF,  p-value: <0.0000000000000002
```

Comments:

Multiple R-squared and Adjusted R-squared are all both 0.975, which is very close to 1. This means that nearly 97.5% part of the relationship between standardized returns of DIA and Ft(the returns of the portfolio) can be explained by this model. Thus means the sd returns of DIA and portfolio are highly correlated. Also, from the figure we know that DJIA and ETF for DJIA are almost identical. Thats why F and the particular market index might be related.

6.Consider the 5 eigenportfolios

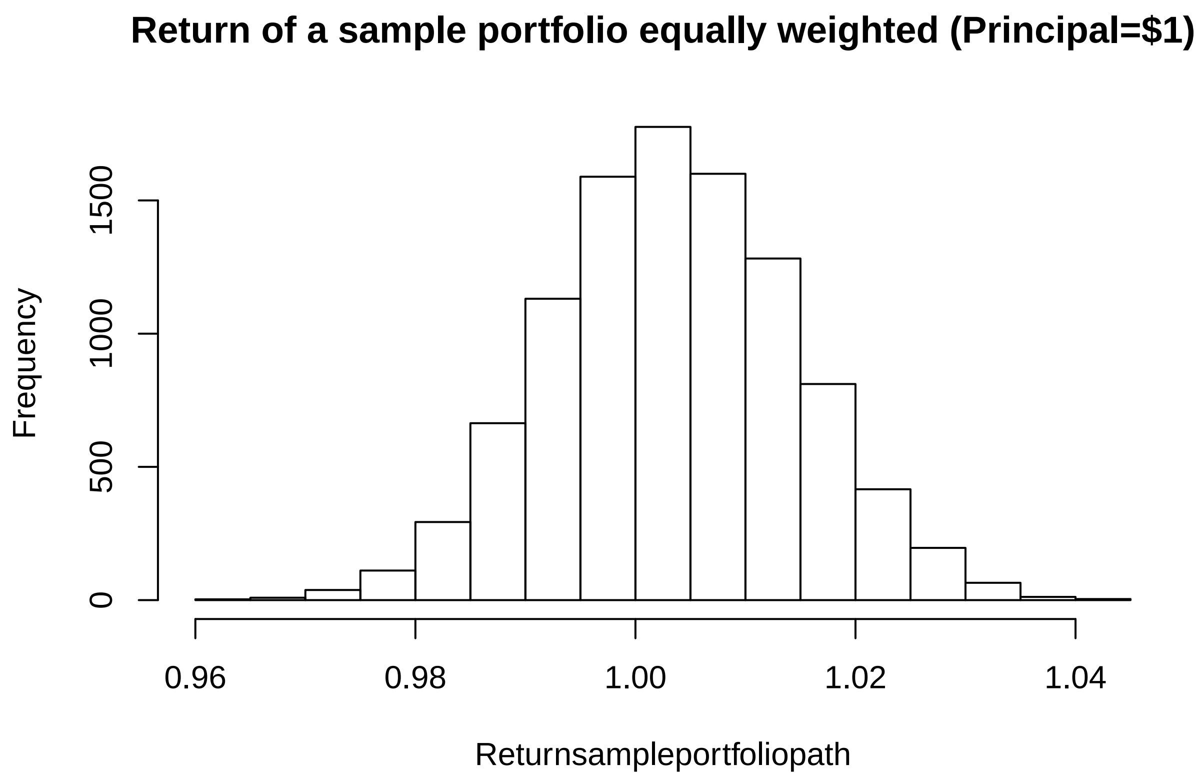
```
#the 5 eigenportfolios(factors)
Ft1 <- as.matrix(R)%*%(eigenvectors[,1]/Std)/sqrt(eigenvalues[1])
Ft2 <- as.matrix(R)%*%(eigenvectors[,2]/Std)/sqrt(eigenvalues[2])
Ft3 <- as.matrix(R)%*%(eigenvectors[,3]/Std)/sqrt(eigenvalues[3])
Ft4 <- as.matrix(R)%*%(eigenvectors[,4]/Std)/sqrt(eigenvalues[4])
Ft5 <- as.matrix(R)%*%(eigenvectors[,5]/Std)/sqrt(eigenvalues[5])
#The standaradized return r should be Y as we calculated in problem 1.
#Y
#Rmean
#Std
```

```r
#Run a regression with the 5 factors and obtain the parameters beta[sk]
beta <- NULL
for (i in 1:30) {
  lm2 <- lm(Y[,i]~ 0+Ft1+Ft2+Ft3+Ft4+Ft5) #the regression intercept should be zero
  beta <- cbind(beta,lm2$coefficients[1:5])
  colnames(beta)[i] <- ticker[i]
}
row.names(beta) <- c("betaFt1","betaFt2","betaFt3","betaFt4","betaFt5")
beta #parameters
```

```
##                WMT         DIS        CAT        XOM        IBM        UNH
## betaFt1 -0.4618736 -0.63094040 -0.6637470 -0.6559986 -0.6291924 -0.6152760
## betaFt2  0.3802979 -0.03605364 -0.3642013 -0.1035184 -0.0480333  0.0239227
## betaFt3 -0.1492199 -0.10338056  0.2364979  0.5407901  0.0361529 -0.1970141
## betaFt4 -0.0582188 -0.00322193 -0.0743655 -0.0460487 -0.1340447  0.2436328
## betaFt5  0.1164549  0.11895748  0.0450171 -0.0848763 -0.1170219 -0.0487225
##                 HD      INTC       AXP        MRK          UTX         MMM
## betaFt1 -0.6671890 -0.628964 -0.633171 -0.5802982 -0.698634866 -0.75120349
## betaFt2  0.0441347 -0.147124 -0.173859  0.2679745 -0.110890211 -0.02341396
## betaFt3 -0.2170395 -0.101303 -0.122624  0.0350672  0.000398858  0.06816428
## betaFt4 -0.0245547 -0.281990  0.293887  0.3456418  0.009660322 -0.03770578
## betaFt5  0.2241929 -0.318229  0.185419 -0.3695749  0.104864533  0.00296111
##                CVX       CSCO      AAPL        MCD        KO          V
## betaFt1 -0.6243715 -0.6965580 -0.558790 -0.5288095 -0.539360 -0.7004037
## betaFt2 -0.1466490 -0.0993503 -0.194085  0.2894640  0.501311 -0.1534246
## betaFt3  0.5580581 -0.1169840 -0.266788 -0.0092882  0.143635 -0.2513417
## betaFt4 -0.0375521 -0.2324379 -0.298775 -0.1606392 -0.229860 -0.0496783
## betaFt5 -0.0508065 -0.2182813 -0.182614  0.1776358  0.169731 -0.0302548
##                WBA        JNJ        PFE      MSFT         PG        JPM
## betaFt1 -0.5060678 -0.6629337 -0.6247861 -0.683732 -0.5396965 -0.7657900
## betaFt2  0.1592517  0.3453508  0.1898118 -0.105794  0.4891567 -0.2431175
## betaFt3 -0.2244960  0.0638118 -0.0806711 -0.221358  0.1244178  0.0600511
## betaFt4  0.1979364  0.1722294  0.3853830 -0.299600 -0.2302360  0.2483929
## betaFt5 -0.0349914 -0.1778019 -0.3397118 -0.236310  0.0704056  0.1462132
##                 VZ       DWDP         GS          BA         NKE
## betaFt1 -0.5003722 -0.6623671 -0.7427119 -0.65976159 -0.54919615
## betaFt2  0.4121287 -0.2436367 -0.2992223 -0.18386466 -0.00419342
## betaFt3  0.1864048  0.1919494  0.0202436  0.00119013 -0.32273090
## betaFt4 -0.0349959 -0.0247386  0.2468770 -0.04388103 -0.06110612
## betaFt5  0.0129552  0.0459421  0.1332894  0.17992529  0.29015026
##                TRV
## betaFt1 -0.6787698
## betaFt2  0.1479617
## betaFt3  0.0926251
## betaFt4  0.1010919
## betaFt5  0.2126418
```

```r
#Calculate the return of a sample portfolio equally weighted in its components.
#First step,generating the path
sampleportfolio <- vector("numeric",length = 30)
sampleportfolio7days <- vector("numeric",length = 30)
sampleportfoliopath <- NULL
sampleportfoliopath7days <- NULL
for (i in 1:10000) {
```

```
  for (j in 1:30) {
    sampleR <- matrix(Rmean[j],10,1)+
      Std[j]*(matrix(rt(10*5,df=3.5),10,5))%*%(beta[,j])+
      Std[j]*sqrt(1-sum(beta[,j]^2))*matrix(rt(10,df=3.5),10,1)
    sampleportfolio[j] <- prod(1+sampleR[1:10,])
    sampleportfolio7days[j] <- prod(1+sampleR[1:7,])
  }
  sampleportfoliopath <- rbind(sampleportfoliopath,sampleportfolio)
  sampleportfoliopath7days <- rbind(sampleportfoliopath7days,sampleportfolio7days)
}
#Second step,calculate the return of portfolio equally weighted in its components
#Assuming principal is $1 and all equally weighted
Returnsampleportfoliopath <- sampleportfoliopath%*%matrix(1/30,30,1)
#histgram of Final 10 days return
hist(Returnsampleportfoliopath,main="Return of a sample portfolio equally weighted (Principal=$1)")
```




```
summary(Returnsampleportfoliopath)
```

```
##        V1
##  Min.   :0.961
##  1st Qu.:0.996
##  Median :1.003
##  Mean   :1.003
##  3rd Qu.:1.011
##  Max.   :1.045
```

```r
#Calculate one week 99% VAR(7days)
Returnsampleportfoliopath7days <- sampleportfoliopath7days%*%matrix(1/30,30,1)
var.99 <- mean(Returnsampleportfoliopath7days) - quantile(Returnsampleportfoliopath7days,0.01)
var.99
```

```
##        1%
## 0.0213428
```

```r
#one week CVAR(7days)
cvar.99 <- mean(Returnsampleportfoliopath7days)-sum(sort(Returnsampleportfoliopath7days)[1:100])/100
cvar.99
```

```
## [1] 0.0251155
```

Bonus Problem

According to research Paper,apply Re-scaled Range(R/S) method and Detrended Fluctuation Analysis (DFA) method. For this problem, I download "EUR/USD exchange rate"" for intraday at 2018.12.03. And the TimeFrame is M1 (1 Minute Bar) Data. #Apply Re-scaled Range(R/S) method

```r
setwd("C:\\Users\\fukaeri\\Desktop\\Stevens\\18FALL\\FE621\\HW")
rate <- read.csv("DAT_MT_EURUSD_M1_201812.csv",head=TRUE,sep=",") #EUR/USD exchange rate
rate <- rate[419:1857,1:6]
nrow(rate)
```

```
## [1] 1439
```

```r
head(rate)
```

```
##     X2018.12.02 X17.00 X1.135000 X1.135000.1 X1.134960 X1.134960.1
## 419  2018.12.03  00:00   1.13501     1.13506   1.13498     1.13503
## 420  2018.12.03  00:01   1.13504     1.13504   1.13496     1.13498
## 421  2018.12.03  00:02   1.13499     1.13512   1.13497     1.13511
## 422  2018.12.03  00:03   1.13511     1.13511   1.13501     1.13501
## 423  2018.12.03  00:04   1.13500     1.13500   1.13489     1.13497
## 424  2018.12.03  00:05   1.13496     1.13502   1.13496     1.13502
```

```r
tail(rate)
```

```
##      X2018.12.02 X17.00 X1.135000 X1.135000.1 X1.134960 X1.134960.1
## 1852  2018.12.03  23:54   1.13786     1.13786   1.13777     1.13780
## 1853  2018.12.03  23:55   1.13778     1.13778   1.13775     1.13777
## 1854  2018.12.03  23:56   1.13776     1.13776   1.13768     1.13770
## 1855  2018.12.03  23:57   1.13776     1.13777   1.13766     1.13767
## 1856  2018.12.03  23:58   1.13768     1.13778   1.13762     1.13776
## 1857  2018.12.03  23:59   1.13776     1.13787   1.13776     1.13777
```

```r
#Apply Re-scaled Range(R/S) method


#Rearrange the data
rate1 <- rate[,3:6]
ratedata <- as.vector(t(as.matrix(rate1)))
head(ratedata) #This is the EUR/USD exchange rate for intraday at 2018.12.03.
```

```
## [1] 1.13501 1.13506 1.13498 1.13503 1.13504 1.13504
```

```
#Step 1.
M <- c()
for (i in 1:(length(ratedata)-1)) {
  M[i] <- log(ratedata[i+1]/ratedata[i])
}
NumberM <- length(M)
NumberM
```

```
## [1] 5755
```

```
#Step 2.
#Since the total number of data observation is 5755
#The only possible value for n is 5 or 1151

#----------------------
#Scenario 1
#divided into 1151(m) sub-series of length 5(n)
n <- 5
m <- ceiling(NumberM/n)
fill <- c(M,rep(mean(M),(n*m-NumberM)))
L <- matrix(fill,nrow = n,ncol = m)
#Step 3.
Z <- apply(L,2,function(x)mean(x))
#Step 4.
C <- matrix(NA,nrow = n,ncol = m)
for (j in 1:m) {
  C[,j] <- L[,j]-Z[j]
}
CD <- apply(C,2,function(x)cumsum(x))
#Step 5
R <- c()
for (i in 1:m) {
  R[i] <- max(C[,i])-min(C[,i])
}
#Step 6
Std <- c()
for (i in 1:m) {
  Std[i] <- sqrt((1/n)*sum(C^2))
}
#Step 7.
R.S1 <- sum(R/Std)/m #R/S Ratio
R.S1
```

```
## [1] 0.0655285
```

```
#----------------------
#Scenario 2
#divided into 5(m) sub-series of length 1151(n)
n <- 1151
m <- ceiling(NumberM/n)
fill <- c(M,rep(mean(M),(n*m-NumberM)))
L <- matrix(fill,nrow = n,ncol = m)
#Step 3.
Z <- apply(L,2,function(x)mean(x))
#Step 4.
```

```r
C <- matrix(NA,nrow = n,ncol = m)
for (j in 1:m) {
  C[,j] <- L[,j]-Z[j]
}
CD <- apply(C,2,function(x)cumsum(x))
#Step 5
R <- c()
for (i in 1:m) {
  R[i] <- max(C[,i])-min(C[,i])
}
#Step 6
Std <- c()
for (i in 1:m) {
  Std[i] <- sqrt((1/n)*sum(C^2))
}
#Step 7.
R.S2 <- sum(R/Std)/m #R/S Ratio
R.S2
```

```
## [1] 5.65259
```

```r
#----------------
#Now fit linear regression
#Step 8&9
R.S <- c(R.S1,R.S2)
R.Sn <- c(5,1151)
lm <- lm(log(R.S)~log(R.Sn))
summary(lm)
```

```
##
## Call:
## lm(formula = log(R.S) ~ log(R.Sn))
##
## Residuals:
## ALL 2 residuals are 0: no residual degrees of freedom!
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    -4.04         NA      NA       NA
## log(R.Sn)       0.82         NA      NA       NA
##
## Residual standard error: NaN on 0 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:    NaN
## F-statistic:   NaN on 1 and 0 DF,  p-value: NA
```

```r
lm$coefficients
```

```
## (Intercept)   log(R.Sn)
##   -4.044253    0.819531
```

We can see that H(beta) is 0.8195305. According to the paper, for data series with long memory effects, H would lie between 0.5 and 1, or elements of the observation are dependent. This means that our "EUR/USD exchange rate" data series for intraday at 2018.12.03 has long memory effects.

Apply Detrended Fluctuation Analysis (DFA) method

```
yt <- cumsum(abs(M))
yt.rev <- rev(yt)
lengthyt <- length(yt)
#Since the total number of data observation is 5755
#The only possible value for n is 5 or 1151

#----------------------
#Scenario 1
#divided into 1151(m) sub-series of length 5(n)
n <- 5
m <- ceiling(NumberM/n)
L <- matrix(yt,nrow = n,ncol = m)
Z <- apply(L,2,function(x)mean(x))
#Fit a linear regression yn(t)
t <- c(1:m)
lmyn1 <- lm(Z~t)
coef1<- lmyn1$coefficients
ynt.1 <- coef1[2]*t+coef1[1]

#Fit a reverse linear regression yn(t)
L.rev <- matrix(yt.rev,nrow = n,ncol = m)
Z.rev <- apply(L.rev,2,function(x)mean(x))
lmyn1.rev <- lm(Z.rev~t)
coef1.rev<- lmyn1.rev$coefficients
ynt.1.rev <- coef1.rev[2]*t+coef1.rev[1]
#Finally the root mean square fluctuation is calculated
Fn1 <- sqrt((1/2*lengthyt)*sum((Z-ynt.1)^2+(Z.rev-ynt.1.rev)^2))
Fn1
```

```
## [1] 52.6692
```

```
#---------------------
#Scenario 2
#divided into 5(m) sub-series of length 1151(n)
n <- 1151
m <- ceiling(NumberM/n)
L <- matrix(yt,nrow = n,ncol = m)
Z <- apply(L,2,function(x)mean(x))
#Fit a linear regression yn(t)
t <- c(1:m)
lmyn1 <- lm(Z~t)
coef1<- lmyn1$coefficients
ynt.1 <- coef1[2]*t+coef1[1]

#Fit a reverse linear regression yn(t)
L.rev <- matrix(yt.rev,nrow = n,ncol = m)
Z.rev <- apply(L.rev,2,function(x)mean(x))
lmyn1.rev <- lm(Z.rev~t)
coef1.rev<- lmyn1.rev$coefficients
ynt.1.rev <- coef1.rev[2]*t+coef1.rev[1]
#Finally the root mean square fluctuation is calculated
Fn2 <- sqrt((1/2*lengthyt)*sum((Z-ynt.1)^2+(Z.rev-ynt.1.rev)^2))
```

```r
Fn2
```

```
## [1] 3.21285
```

```r
#-----------------
#Now fit linear regression between F(n) and n
Fn <- c(Fn1,Fn2)
Fn.n <- c(1151,5)
lm.Fn <- lm(log(Fn)~log(Fn.n))
summary(lm.Fn)
```

```
##
## Call:
## lm(formula = log(Fn) ~ log(Fn.n))
##
## Residuals:
## ALL 2 residuals are 0: no residual degrees of freedom!
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    0.340         NA      NA       NA
## log(Fn.n)      0.514         NA      NA       NA
##
## Residual standard error: NaN on 0 degrees of freedom
## Multiple R-squared:     1,   Adjusted R-squared:   NaN
## F-statistic:  NaN on 1 and 0 DF,  p-value: NA
```

```r
lm.Fn$coefficients
```

```
## (Intercept)   log(Fn.n)
##    0.339537    0.514230
```

We can see that the slope is 0.5142304, which indicates that data series is with long-range power law correlations.