Assignment

FE630 - PORTFOLIO THEORY AND APPLICATION

Napat Loychindarat
05/18/2018 |

1.

```r
#Set the directory that data is being contained.
setwd("C:\\Users\\nackz\\Desktop\\Stevens Institute\\Subjects\\FE630 - Portfolio Theory and Ap
plications\\Midterm\\data")
require("knitr")

## Loading required package: knitr

## Warning: package 'knitr' was built under R version 3.4.2

opts_knit$set(root.dir = "C:\\Users\\nackz\\Desktop\\Stevens Institute\\Subjects\\FE630 - Port
folio Theory and Applications\\Midterm\\data")

file.list <- list.files()

for(i in 1:length(file.list)){
    pos <- gregexpr(".txt", file.list[i])[[1]]
    assign(substr(file.list[i], 1, pos-1), read.table(file.list[i], header = FALSE))
}
```

2.and 3.

```r
library(xts)

## Loading required package: zoo

##
## Attaching package: 'zoo'

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

library(quantmod)

## Loading required package: TTR

## Warning: package 'TTR' was built under R version 3.4.2

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

#Initilze matrix P and R to contain the data
P <- NULL
R <- NULL

#Run the loop the following the number of files
for(i in 1:length(file.list)){

    #Use read.table function to read the data in .txt format.
    data <- read.table(file.list[i], header = FALSE)
    #Convert date from text format into date format.
    data$V1 <- as.Date(as.character(data$V1), format = "%Y%m%d")

    #if it is the first file, then take date into matrices
    if(i == 1){
        P <- data[,c(1,7)]
        #set R matrix to be time-series type.
        R <- xts(data[,7], order.by=data[,1])
        #Calculate simple return using quantmod library.
        R <- dailyReturn(R)
    }else{
```

```r
        P <- cbind(P, data[,7])
        data <- xts(data[,7], order.by=data[,1])
        #After first file, calculate simeple return and combine to the matrix
        R <- cbind(R, dailyReturn(data))
    }
}

#Set column name of both matrices.
colnames(P) <- c("date", gsub(".txt", "", file.list))
colnames(R) <- gsub(".txt", "", file.list)
```

4.

```r
#Calculate the means of each company from R Matrix (return matrix)
mu <- colMeans(R)
mu

##            AA           AXP            BA           BAC           CAT
##  2.249995e-04  4.719190e-04  9.440651e-04  5.397095e-04 -3.399291e-04
##          CSCO           CVX            DD           DIS            GE
##  5.625233e-04 -2.946526e-04  3.384351e-04  1.097887e-03  4.246304e-04
##            HD           HPQ           IBM          INTC           JNJ
##  1.031994e-03  1.008988e-03 -2.700954e-04  6.676937e-04  5.202620e-04
##           JPM            KO           MCD           MMM           MRK
##  6.361132e-04  2.400689e-04  2.896139e-04  7.037235e-04  4.671944e-04
##          MSFT           PFE            PG             T           TRV
##  8.997502e-04  4.902265e-04  2.328887e-04  1.386966e-04  5.944964e-04
##           UNH           UTX            VZ           WMT           XOM
##  1.253373e-03  1.993582e-04  2.252429e-04  2.158076e-05 -1.064571e-04
```

5.

```r
#Calculate the covariance from R Matrix(return matrix)
Q <- cov(R)
```

6.

```r
#Export the mu vector and Q matrix by the name of inputs.RData.
save(mu, Q, file="inputs.RData")
```

Question 2

$$max\left[U(h) = -\frac{1}{2}h^T Q h + \tau h^T \mu\right]$$

$$subjust\ to\ A_{eq}\omega = b_{eq}$$

$$with A_{eq} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & : & -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & : & 0 & -1 & 0 & 0 & 0 \\ : & : & : & : & : & : & : & : & : & : & : & : \\ : & : & : & : & : & : & : & : & : & : & : & : \\ 1 & 0 & 0 & 0 & 1 & 0 & : & 0 & 0 & 0 & -1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 & : & 0 & 0 & 0 & 0 & -1 \end{bmatrix}_{30x61}$$

$$b_{eq} = [1 \quad 0 \quad \cdots \quad 0 \quad -0.1 \quad \cdots \quad -0.1]_{1x61}$$

$$\mathrm{d}_{vector} = [\tau\mu]_{1x30}$$

```
library(quadprog)

port <- function(mu, Q, tau){

    Dmat <- Q
    dvec <- matrix(tau*mu, nrow = nrow(Q), ncol = 1)
    Amat <- cbind(matrix(rep(1,nrow(Q)), ncol=1), diag(nrow(Q)), -diag(nrow(Q)))
    bvec <- c(1, rep(0,nrow(Q)), rep(-0.1, nrow(Q)))

    result <- solve.QP(Dmat, dvec, Amat, bvec, meq=2)
    result$solution

    return(result$solution)
}
```

Question 3

1.

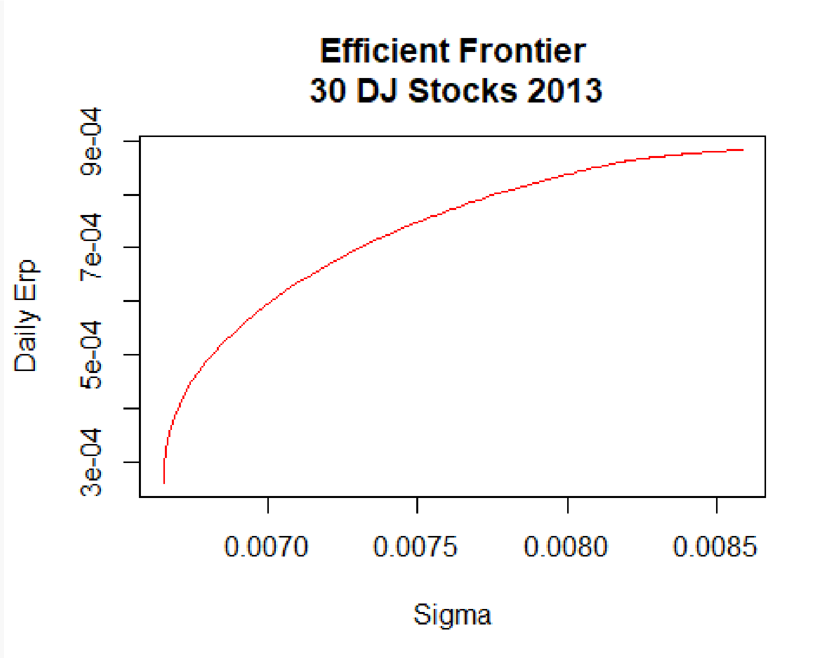
```
load("inputs.RData")
```

2.

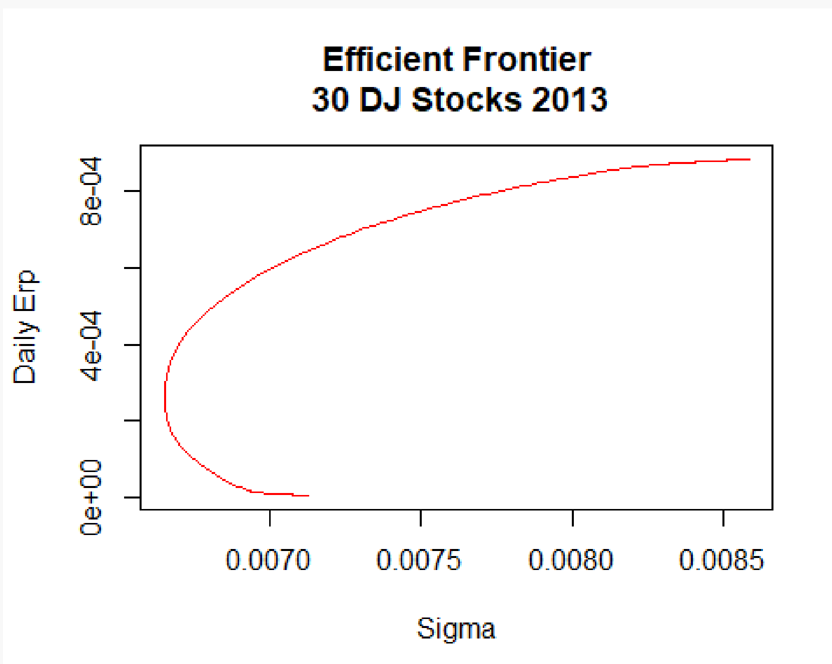
```
TAU <- seq(0, 0.5, 0.001)
```

3. 4. 5. And 6.

```
#Put the 'port' function in the TAU loop and keep the optimal weights from each tau.
w <- NULL
for(i in TAU){
    w <- rbind(w, port(mu, Q, i))
}
w <- matrix(w, ncol=30)

#Create 'eff.frontier' function to calculate mean and variance of each optimum portfolio.
eff.frontier <- function(mu, Q, tau=seq(0, 0.5, 0.001), port){

    w <- NULL
    for(i in tau){
        w <- rbind(w, port(mu, Q, i))
    }
    w <- round(w, 10)
    Erp <- NULL
    Sigma <- NULL
    for(i in 1:nrow(w)){
        Erp <- rbind(Erp, t(matrix(w[i,]))%*%matrix(mu))
        Sigma <- rbind(Sigma, sqrt(t(matrix(w[i,]))%*%Q%*%matrix(w[i,])))
    }
    return(data.frame(Erp = Erp, Sigma = Sigma))
}
portf <- eff.frontier(mu, Q, TAU, port)

#Calculate the optimal sharpe ration to find the optimal tau and no.79 is the optimal oone.
plot(portf$Sigma , portf$Erp, col="red", main="Efficient Frontier\n 30 DJ Stocks 2013", xlab =
"Sigma", ylab = "Daily Erp", type = "l")
points(portf$Sigma[which.max(portf$sharpe)], portf$Erp[which.max(portf$sharpe)],
       col="blue", pch=20)
```




```
portf <- eff.frontier(mu, Q, tau=seq(-5, 5, 0.001), port)


#When we increase tau, the graph become as below.
plot(portf$Sigma , portf$Erp, col="red", main="Efficient Frontier\n 30 DJ Stocks 2013", xlab =
"Sigma", ylab = "Daily Erp", type = "l")
points(portf$Sigma[which.max(portf$sharpe)], portf$Erp[which.max(portf$sharpe)],
       col="blue", pch=20)
```



Question 4

1.

```
setwd("C:\\Users\\nackz\\Desktop\\Stevens Institute\\Subjects\\FE630 - Portfolio Theory and Ap
plications\\Midterm\\Midtermdata")

opts_knit$set(root.dir = "C:\\Users\\nackz\\Desktop\\Stevens Institute\\Subjects\\FE630 - Port
folio Theory and Applications\\Midterm\\Midtermdata")

load("data.rda")
```

2.

```
#Use quantmod library to calculate daily simple return.
library(quantmod)

prices <- data.frame(prices)

date <- as.Date(as.character(row.names(prices)), format = "%Y%m%d")
prices <- xts(prices[,c(1:30)], order.by=date)

colname <- colnames(prices)
Prices <- NULL

for(i in colname){
    Prices <- cbind(Prices, dailyReturn(prices[,i]))

}
colnames(Prices) <- colname
```

3.

```
#Annualize the return by 252.
Prices <- Prices*252
```

4.

```
#Move the index to separate vector.
index.data <- Prices[-1,1]
Prices <- Prices[-1,-1]
```

5.

```
#Calculate the covariance.
Qts <- cov(Prices)
```

6.

```
#Print out the first row and first five columns of covariance matrix.
head(Qts[,1:5], 5)

##           AAPL       AXP        BA       CAT      CSCO
## AAPL 17.189758  5.163950  8.117157  7.504042  7.760136
## AXP   5.163950 11.801005  3.709621  5.023068  3.464257
## BA    8.117157  3.709621 11.578327  5.912234  6.436918
## CAT   7.504042  5.023068  5.912234 15.586600  6.221941
## CSCO  7.760136  3.464257  6.436918  6.221941 12.601319
```

Question 5

1.

```r
#Use 'lm' function to run linear regression and get intercept, beta and SD of residual σ_{R_i}.
#Keep all values the table.
table5 <- NULL

for(i in 1:ncol(Prices)){

    #Run linear regression in R: lm() function.
    b <- lm(Prices[,i] ~ index.data[,1])
    #Get Y-intercept from the result to be our alpha.
    yi <- summary(b)$coefficients[1]
    #Get slope from the result to be our beta.
    bi <- summary(b)$coefficients[2]
    #Get all residual to compute sd.
    sigma.ri <- sd(summary(b)$residuals)
    table5 <- rbind(table5, c(yi,bi,sigma.ri))

}

table5 <- data.frame(table5)
colnames(table5) <- c("intercept", "Bi", "sigmaRi")
row.names(table5) <- colname[-1]
```

2.

```r
#Print out table 5. The columns are Y-intercept, slope (βi) and idiosyncratic deviation
table5

##        intercept        Bi  sigmaRi
## AAPL  0.06744403 1.1387824 3.143631
## AXP  -0.18112491 0.8535259 2.774170
## BA    0.17556675 1.0196946 2.391526
## CAT  -0.32505106 1.0635297 3.035311
## CSCO  0.18896215 1.0600683 2.503847
## CVX  -0.25978829 1.1483061 3.220531
## DD   -0.04566343 1.0205799 2.850854
## DIS   0.23693452 0.9171245 2.605526
## GE    0.15146337 1.0652192 2.518691
## GS   -0.01794989 1.1652760 1.741245
## HD    0.25281538 1.0155424 1.973821
## IBM  -0.13038643 1.0502861 2.188652
## INTC  0.03174961 0.9995495 3.233781
## JNJ  -0.05511518 0.8540402 1.653148
## JPM   0.06634039 1.2041648 1.809290
## KO    0.04972505 0.6628241 1.667704
## MCD   0.20206365 0.9300115 2.271137
## MMM   0.02914607 0.9639493 1.529654
## MRK  -0.05239646 0.9530091 2.577770
## MSFT  0.15874807 1.2436208 3.384866
## NKE   0.34645997 0.9820348 2.422494
## PFE   0.15625672 0.8856754 2.055044
## PG   -0.10495667 0.7662929 1.639514
## TRV   0.12164608 0.8736200 1.652621
## UNH   0.22330586 1.0310370 2.975608
## UTX  -0.06055893 0.9770890 2.105933
## V     0.36030805 1.1184232 2.592784
## VZ   -0.04878761 0.7487452 1.818478
```

```
## WMT  -0.26964275 0.7750674 2.723288
## XOM  -0.14942583 1.0512685 2.406803
```

3.

```
#Compute the variance of DJI.
var.market <- var(index.data)
var.market

##          X.DJI
## X.DJI 5.612174
```

4.

```
#Using variance of index, beta and variance of idiosyncratic variance.
Qsi <- as.numeric(var.market)*t(matrix(table5$Bi, nrow = 1))%*%(matrix(table5$Bi, nrow = 1)) +
diag(table5$sigmaRi^2)
```

5.

```
#Print the first five rows and columns of covariance and omega.
head(Qsi[,1:5], 5)

##           [,1]      [,2]      [,3]      [,4]      [,5]
## [1,] 17.189758  5.476909  6.543181  6.824462  6.802251
## [2,]  5.476909 11.801005  4.904163  5.114985  5.098338
## [3,]  6.543181  4.904163 11.578327  6.110796  6.090908
## [4,]  6.824462  5.114985  6.110796 15.586600  6.352746
## [5,]  6.802251  5.098338  6.090908  6.352746 12.601319

omega <- diag(Qsi)
omega <- diag(omega, nrow = nrow(Qsi), ncol = ncol(Qsi))
head(omega[,1:5], 5)

##          [,1]   [,2]     [,3]    [,4]     [,5]
## [1,] 17.18976  0.000  0.00000  0.0000  0.00000
## [2,]  0.00000 11.801  0.00000  0.0000  0.00000
## [3,]  0.00000  0.000 11.57833  0.0000  0.00000
## [4,]  0.00000  0.000  0.00000 15.5866  0.00000
## [5,]  0.00000  0.000  0.00000  0.0000 12.60132
```

Question 6

1.

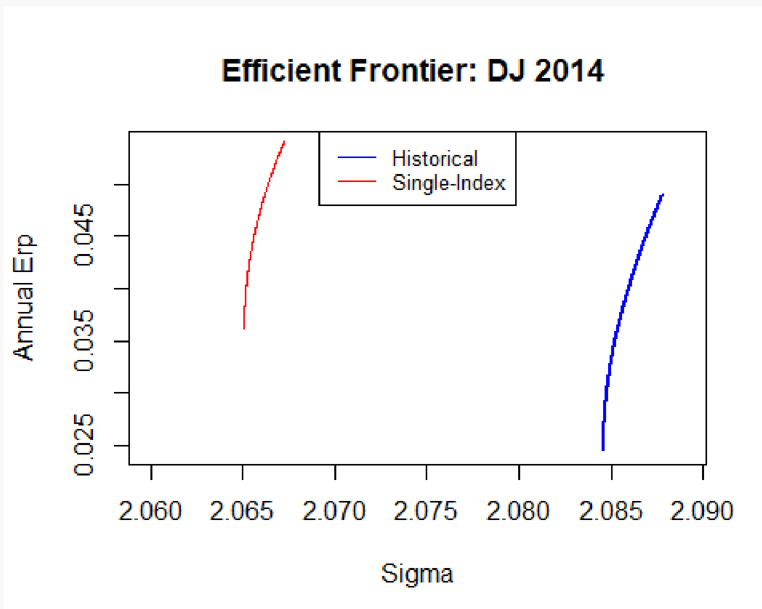
```r
#Use ‘eff.frontier’ function which contains ‘port’ function to get new efficient frontier by Q
TS convenience.
port2 <- eff.frontier(as.matrix(colMeans(Prices), nrow=1) , Qts, TAU,port=port)
```

2.

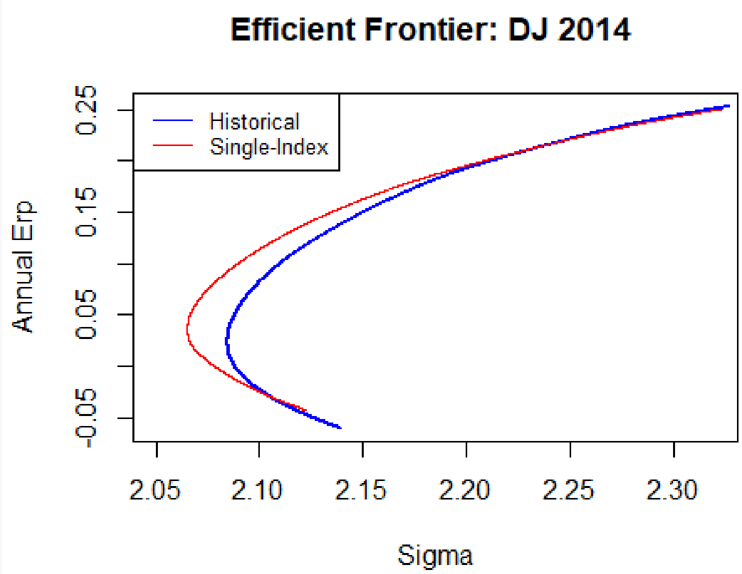
```r
#Use ‘eff.frontier’ function which contains ‘port’ function to get new efficient frontier by Q
SI covariance.
port3 <- eff.frontier(as.matrix(colMeans(Prices), nrow=1) , Qsi,TAU, port=port)
```

3.

```r
#Plot and compare 2 graphs
#As we can clearly see, 2 efficient frontiers are very different. We may be able to conclude t
hat Single-Index model is not valid.
plot(port2$Sigma , port2$Erp, col="blue", main="Efficient Frontier: DJ 2014 ", xlab = "Sigma",
ylab = "Annual Erp", type = "l", lwd = 2, ylim=c(0.02440589,0.05377408), xlim=c(2.06,2.089))
lines(port3$Sigma , port3$Erp, col="red")
legend("top", c("Historical","Single-Index"),
       col=c("blue","red"), lwd = c(1,1),cex=0.8)
```




```r
port4 <- eff.frontier(as.matrix(colMeans(Prices), nrow=1) , Qts, seq(-5, 5, 0.001),port=port)
port5 <- eff.frontier(as.matrix(colMeans(Prices), nrow=1) , Qsi, seq(-5, 5, 0.001),port=port)


#However, when we increase the range of TAU, we get the curves very close each other when we i
ncrease tau which affects mu. We can probably conclude that single-index model is valid when w
e have large enough mu.
plot(port4$Sigma , port4$Erp, col="blue", main="Efficient Frontier: DJ 2014 ", xlab = "Sigma",
ylab = "Annual Erp", type = "l", lwd = 2, xlim=c(2.05,2.32))
lines(port5$Sigma , port5$Erp, col="red")
legend("topleft", c("Historical","Single-Index"),
       col=c("blue","red"), lwd = c(1,1),cex=0.8)
```


Efficient Frontier: DJ 2014
Historical
Single-Index
Annual Erp
0.25
0.15
0.05
-0.05
2.05
2.10
2.15
2.20
2.25
2.30
Sigma