Project 4

Juan Piao

4/27/2020

Please answer the following questions assuming the single index model holds:

a. Use your project data in the period 01-Jan-2012 to 01-Jan-2017

```
#Read your csv file:
a <- read.csv("stockData.csv", sep=",", header=TRUE)
#Convert adjusted close prices into returns:
r1 <- (a[-1,3:ncol(a)]-a[-nrow(a),3:ncol(a)])/a[-nrow(a),3:ncol(a)]
```

1. Compute estimates for α_i, β_i, σ_{ϵ_i} , i = 1, 2, . . . , 30 by regressing each stock's return on the S&P 500.

```
#regressing each stock's return on the S&P 500.
beta1 <- rep(0,30)

alpha1 <- rep(0,30)

sigma_e1 <- rep(0,30)

var_beta1 <- rep(0,30)

for(i in 1:30){
    q <- lm(data=r1, formula=r1[,i+1] ~ r1[,1])
    beta1[i] <- q$coefficients[2]
    alpha1[i] <- q$coefficients[1]
    sigma_e1[i] <- summary(q)$sigma^2
    var_beta1[i] <- vcov(q)[2,2]
}

cbind(beta1, alpha1, sigma_e1)
```

```
##            beta1        alpha1     sigma_e1
##  [1,] 1.24074863  0.0021365066 0.0039452927
##  [2,] 1.37375480 -0.0009731452 0.0031363465
##  [3,] 1.08178436  0.0066892949 0.0030359930
##  [4,] 1.88423373 -0.0033598495 0.0103459161
##  [5,] 1.04781266  0.0119771730 0.0019528764
##  [6,] 1.90807430  0.0070192694 0.0111573895
##  [7,] 0.65190754  0.0082681527 0.0030831563
##  [8,] 0.27061913  0.0160978305 0.0034753848
##  [9,] 1.16408144 -0.0110756155 0.0065018654
## [10,] 0.39957489  0.0083725850 0.0057450602
## [11,] 0.23703300  0.0136925064 0.0023112820
## [12,]  0.09756465  0.0053284378 0.0018616390
## [13,]  0.80578326  0.0054595028 0.0007791898
## [14,]  1.00062653  0.0130565941 0.0017425193
## [15,]  1.50513175  0.0045568279 0.0027042975
## [16,]  1.68682347 -0.0016018865 0.0034384804
## [17,]  1.25585433  0.0095210098 0.0018103129
## [18,]  1.26241681 -0.0084069724 0.0075358387
## [19,]  0.75455155  0.0056211016 0.0009921343
## [20,]  0.86017308  0.0051505716 0.0016704491
## [21,] -1.75919729  0.0475876935 0.0734132548
## [22,]  0.79808187  0.0066821987 0.0057316492
## [23,]  1.39094779 -0.0025157261 0.0013181984
## [24,]  0.50810408  0.0158926013 0.0040854922
## [25,]  1.39914510  0.0133483415 0.0049070729
## [26,]  0.73909215  0.0047233304 0.0022046421
## [27,]  1.09687284  0.0113593945 0.0012300566
## [28,]  0.43744613  0.0112371367 0.0030827336
## [29,]  1.32518504  0.0335596314 0.0273239688
## [30,]  1.28846628  0.0086291312 0.0024983928
```

2. Construct the 30 $\times$ 30 variance covariance matrix based on the single index model.

```
covmat <- cov(r1)  #With ^GSPC

var_market<-covmat[1,1]
covmat1<- var_market * beta1 %*% t(beta1)
diag(covmat1) <- diag(covmat1)+ sigma_e1

#covmat1
```

3. Answer the same question as in project 2, part (b), question (1) using the new inputs from (1) above. Draw the frontier on the same plot as in project 2. Now you will have two frontiers, one using the historical variance covariance matrix (project 2) and one using the variance covariance matrix with inputs from the single index model.

The efficient frontier obtained by using the historical variance covariance matrix (project 2) and using the new inputs from (1) above is shown below.

```
covmat <- cov(r1)  #With ^GSPC
#Compute mean vector:
means <- colMeans(r1)  #With ^GSPC

#Compute the vector of standard deviations:
stdev <- diag(covmat)^.5

ones <- rep(1, 30)


#Composition of minimum risk portfolio:
x1 <- ( solve(covmat[2:31, 2:31]) %*% ones ) / as.numeric( t(ones) %*% solve(covmat[2:31, 2:31]) %*% on

#Mean:
m1 <- t(x1) %*% means[2:31]
```

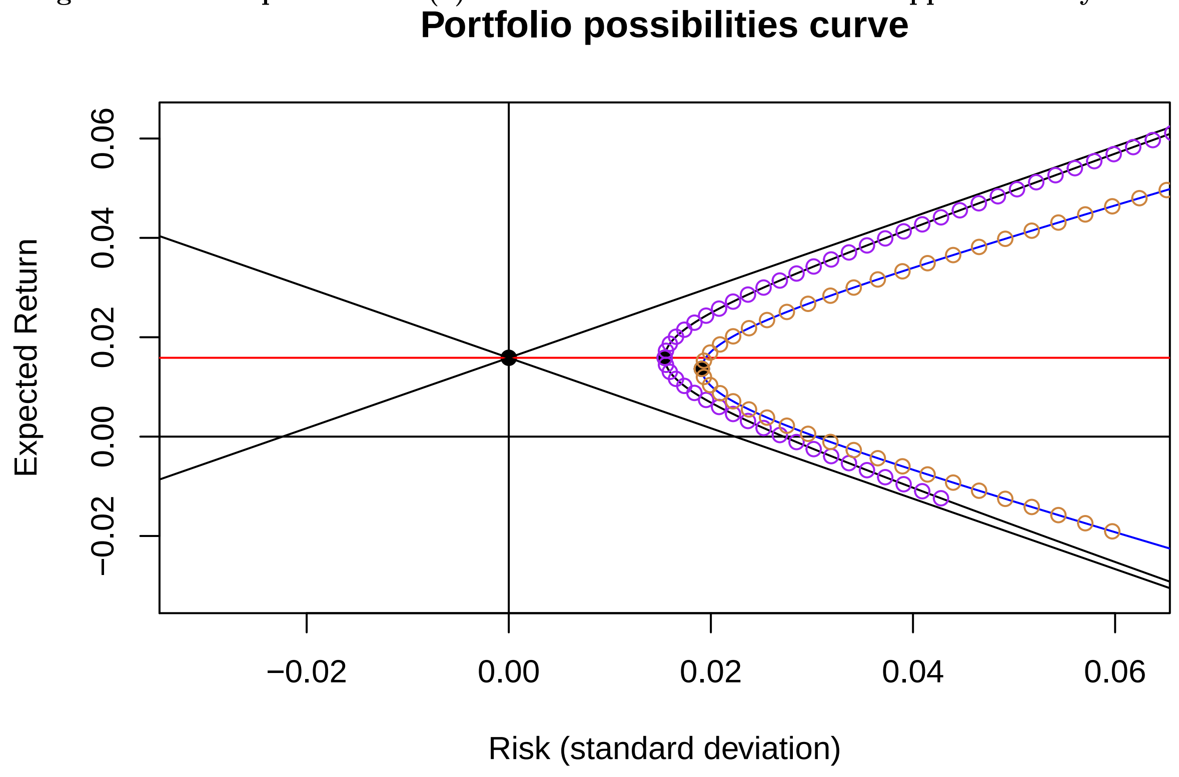
```r
#Variance:
v1 <- t(x1) %*% covmat[2:31, 2:31] %*% x1



#Compute A:
A <- t(ones) %*% solve(covmat[2:31, 2:31]) %*% means[2:31]


#Compute B:
B <- t(means[2:31]) %*% solve(covmat[2:31, 2:31]) %*% means[2:31]


#Compute C:
C <- t(ones) %*% solve(covmat[2:31, 2:31]) %*% ones


#Compute D:
D <- B*C - A^2

#Portfolio 2:  (It doesn't have to be efficient, as long as it is on the frontier).
#Need to choose a value of E.  Let's say, E=0.015.
#To find x2 we use our class notes (see week 2 - lecture 1 notes):
#x2=lambda1*Sigma^-1*means + lambda2*Sigma^-1*ones
#lambda1 = (CE-A)/D and lambda2=(B-AE)/D.

E <- 0.03
lambda1 <- (C*E-A)/D
lambda2 <- (B-A*E)/D

x2=as.numeric(lambda1)*solve(covmat[2:31, 2:31]) %*% means[2:31] +
as.numeric(lambda2)* solve(covmat[2:31, 2:31]) %*% ones

#Mean:
m2 <- t(x2) %*% means[2:31]

#Variance:
v2 <- t(x2) %*% covmat[2:31, 2:31] %*% x2

#We also need the covariance between portfolio 1 and portfolio 2:
cov_ab <- t(x1) %*% covmat[2:31, 2:31] %*% x2

#Now we have two portfolios on the frontier.  We can combine them to trace out the entire frontier:
#Let a be the proportion of investor's wealth invested in portfolio 1.
#Let b be the proportion of investor's wealth invested in portfolio 2.

a <- seq(-3,3,.1)
b <- 1-a

r_ab <- a*m1 + b*m2

var_ab <- a^2*v1 + b^2*v2 + 2*a*b*cov_ab
sd_ab <- var_ab^.5
```

```r
#Give values for E:
E <- seq(-5,5,.1)

#Compute sigma2 as a function of A,B,C,D, and E:
#sigma2 <- (C*E^2 - 2*A*E +B) /D


plot(0, A/C, main = "Portfolio possibilities curve", xlab = "Risk (standard deviation)",
  ylab = "Expected Return", type = "n",
  xlim = c(-2*sqrt(1/C), 4*sqrt(1/C)),
  ylim = c(-2*A/C, 4*A/C))

#Plot center of the hyperbola:
    points(0, A/C, pch = 19)

#Plot transverse and conjugate axes:
    abline(v = 0) #Also this is the y-axis.
    abline(h = A/C, col="red")

#Plot the x-axis:
    abline(h = 0)

#Plot the minimum risk portfolio:
    points(sqrt(1/C), A/C, pch=19)

#Find the asymptotes:
    V <- seq(-1, 1, 0.001)
    A1 <- A/C + V * sqrt(D/C)
    A2 <- A/C - V * sqrt(D/C)
    points(V, A1, type = "l")
    points(V, A2, type = "l")

#Efficient frontier:
    minvar <- 1/C
    minE <- A/C
    sdeff <- seq((minvar)^0.5, 1, by = 0.0001)
    options(warn = -1)
    y1 <- (A + sqrt(D*(C*sdeff^2 - 1)))*(1/C)
    y2 <- (A - sqrt(D*(C*sdeff^2 - 1)))*(1/C)
    options(warn = 0)

    points(sdeff, y1, type = "l")
    points(sdeff, y2, type = "l")


points(sd_ab, r_ab, col="purple")



##################################################################################
# compute A,B,C, D for single index model

#Compute A:
```

```r
A1 <- t(ones) %*% solve(covmat1) %*% means[2: 31]


#Compute B:
B1 <- t(means[2: 31]) %*% solve(covmat1) %*% means[2: 31]


#Compute C:
C1 <- t(ones) %*% solve(covmat1) %*% ones


#Compute D:
D1 <- B1*C1 - A1^2

#Efficient frontier from SIM:
    minvar1 <- 1/C1
    minE1 <- A1/C1
    sdeff1 <- seq((minvar1)^0.5, 1, by = 0.0001)
    options(warn = -1)
    y1 <- (A1 + sqrt(D1*(C1*sdeff1^2 - 1)))*(1/C1)
    y2 <- (A1 - sqrt(D1*(C1*sdeff1^2 - 1)))*(1/C1)
    options(warn = 0)

    points(sdeff1, y1, type = "l", col="blue")
    points(sdeff1, y2, type = "l", col="blue")

#Plot the minimum risk portfolio:
    points(sqrt(1/C1), A1/C1, pch=19)

##### get two points on the frontier

#Composition of minimum risk portfolio:
x3 <- ( solve(covmat1) %*% ones ) / as.numeric( t(ones) %*% solve(covmat1) %*% ones )

#Mean:
m3 <- t(x3) %*% means[2:31]

#Variance:
v3 <- t(x3) %*% covmat1 %*% x3


#Portfolio 2:
#To find x2 we use our class notes (see week 2 - lecture 1 notes):
#x2=lambda1*Sigma^-1*means + lambda2*Sigma^-1*ones
#lambda1 = (CE-A)/D and lambda2=(B-AE)/D.

E <- 0.03
lambda1 <- (C1*E-A1)/D1
lambda2 <- (B1-A1*E)/D1

x4=as.numeric(lambda1)*solve(covmat1) %*% means[2:31] +
as.numeric(lambda2)* solve(covmat1) %*% ones
```

```
#Mean:
m4 <- t(x4) %*% means[2:31]

#Variance:
v4 <- t(x4) %*% covmat1 %*% x4

#We also need the covariance between portfolio 1 and portfolio 2:
cov_ab1 <- t(x3) %*% covmat1 %*% x4


#Now we have two portfolios on the frontier.  We can combine them to trace out the entire frontier:
#Let a be the proportion of investor's wealth invested in portfolio 1.
#Let b be the proportion of investor's wealth invested in portfolio 2.

r_ab1 <- a*m3 + b*m4

var_ab1 <- a^2*v3 + b^2*v4 + 2*a*b*cov_ab1
sd_ab1 <- var_ab1^.5

points(sd_ab1, r_ab1, col="peru")
```

The efficient frontier obtained by using the historical variance covariance matrix (project 2) is in black line overlapped with purple dots. The efficient frontier obtained by using the new inputs from (1) above is in blue line overlapped with yellow dots.



b. Adjusting the betas:

Adjust the betas using Blume's. For the Blume technique use the two periods: 01- Jan-2012 to 01-Jan-2017 and 01-Jan-2017 to 31-Mar-2020.

```
#Read your csv file:
a2 <- read.csv("stockData-2nd.csv", sep=",", header=TRUE)
#Convert adjusted close prices into returns:
r2 <- (a2[-1,3:ncol(a2)]-a2[-nrow(a2),3:ncol(a2)])/a2[-nrow(a2),3:ncol(a2)]
```

```
#Compute the variance covariance matrix of the returns for the 2nd period:
covmat2 <- var(r2)
#Compute the betas in the 2nd period:
beta2 <- covmat2[1,-1] / covmat2[1,1]

#Adjust betas using the Blume's technique:
q1 <- lm(beta2 ~ beta1)

beta3adj_blume <- q1$coef[1] + q1$coef[2]*beta2
```

Adjust the betas using Vasicek 's. For the Vasicek technique use only the period 01-Jan-2012 to 01-Jan-2017.

```
#Vasicek's method:
beta2 <- rep(0,30)

alpha2 <- rep(0,30)

sigma_e2 <- rep(0,30)

var_beta2 <- rep(0,30)

for(i in 1:30){
    q <- lm(data=r1, formula=r1[,i+1] ~ r1[,1])
    beta2[i] <- q$coefficients[2]
    alpha2[i] <- q$coefficients[1]
    sigma_e2[i] <- summary(q)$sigma^2
    var_beta2[i] <- vcov(q)[2,2]
}


#Adjusting the betas using the Vasicek's technique:
beta2adj_vasicek <- var_beta2*mean(beta2)/(var(beta2)+var_beta2) +
var(beta2)*beta2/(var(beta2)+var_beta2)
```

```
#Now let's compare:
#Note:
#beta2:  Actual betas in period 2.
#beta3adj_blume:  Adjusted betas (Blume) that can be used as forecast
#                 for period 3.
#beta2adj_vasicek:  Adjusted betas (Vasicek) that can be used as forecast for period 3.

cbind(beta2, beta3adj_blume, beta2adj_vasicek)
```

```
##              beta2 beta3adj_blume beta2adj_vasicek
## AAPL   1.24074863      0.9279823        1.1969108
## CSCO   1.37375480      0.8744099        1.3228394
## MSFT   1.08178436      0.8609328        1.0644134
## SNE    1.88423373      0.8808086        1.5997427
## ADBE   1.04781266      0.8980639        1.0386830
## MU     1.90807430      1.0448517        1.6007929
## CWT    0.65190754      0.6951816        0.6822026
## SJW    0.27061913      0.7556570        0.3515179
## NRG    1.16408144      0.8379889        1.1138142
## KEP    0.39957489      0.8105236        0.4988702
## NEE    0.23703300      0.7456802        0.2960493
## SO     0.09756465      0.8017888        0.1557261
## BRK.B  0.80578326      0.8792969        0.8094077
## V      1.00062653      0.8852470        0.9955384
## JPM    1.50513175      0.9872436        1.4475152
## C      1.68682347      1.1494880        1.5931891
## MA     1.25585433      0.9252018        1.2330760
## LFC    1.26241681      0.9704317        1.1829596
## JNJ    0.75455155      0.8288919        0.7611155
## CVS    0.86017308      0.8504082        0.8642187
## ISRG  -1.75919729      0.9137449        0.2507161
## NVO    0.79808187      0.7634198        0.8218433
## ABT    1.39094779      0.8151731        1.3671710
## CI     0.50810408      0.8475286        0.5673375
## AMZN   1.39914510      0.9205801        1.3200422
## TM     0.73909215      0.8249519        0.7542900
## HD     1.09687284      0.9004661        1.0886235
## NKE    0.43744613      0.8945785        0.4916343
## TSLA   1.32518504      0.8383827        1.1138727
## LOW    1.28846628      0.9958214        1.2548202
```

```
#PRESS1 <- sum((beta2-beta3)^2) / 30


PRESS3 <- sum((beta2adj_vasicek - beta2)^2) / 30
```

```
PRESS3
```

Compute PRESS only for the Vasicek technique.

```
## [1] 0.1441885
```