Project 6

Yaman Yucel

2023-05-05

Part a: Use only the stocks with positive betas in your data. Rank the stocks based on the excess return to beta ratio and complete the entire table based on handout #28

```
#Part a
#Read all data
a_all <- read.csv("stockData.csv", sep=",", header=TRUE)

# Use 5 year data to train
a <- a_all[1:60,]

#Convert adjusted close prices into returns:
r <- (a[-1,3:ncol(a)]-a[-nrow(a),3:ncol(a)])/a[-nrow(a),3:ncol(a)] # return of stocks + market
#r_m <- (a[-1,3]-a[-nrow(a),3])/a[-nrow(a),3] # return of market

n_stocks = 30

covmat <- var(r)
beta <- covmat[1,-1]/ covmat[1,1]

rrr <- r[,-c(1,which(beta<0)+1)]


beta <- rep(0,ncol(rrr))
alpha <- rep(0,ncol(rrr))
mse <- rep(0,ncol(rrr))
Ribar <- rep(0,ncol(rrr))
Ratio <- rep(0,ncol(rrr))
stock <- rep(0,ncol(rrr))

rf <- 0.002


for(i in 1:ncol(rrr)){
  q <- lm(data=rrr, formula=rrr[,i] ~ r[,1])
  beta[i] <- q$coefficients[2]
  alpha[i] <- q$coefficients[1]
  mse[i] <- summary(q)$sigma^2
  Ribar[i] <- q$coefficients[1]+q$coefficients[2]*mean(r[,1])
  Ratio[i] <- (Ribar[i]-rf)/beta[i]
  stock[i] <- i
}

xx <- (cbind(stock,alpha, beta, Ribar, mse, Ratio))
```

```r
A <- xx[order(-xx[,6]),]

col1 <- rep(0,nrow(A))
col2 <- rep(0,nrow(A))
col3 <- rep(0,nrow(A))
col4 <- rep(0,nrow(A))
col5 <- rep(0,nrow(A))

col1 <- (A[,4]-rf)*A[,3]/A[,5]
col3 <- A[,3]^2/A[,5]
for(i in(1:nrow(A))) {
  col2[i] <- sum(col1[1:i])
  col4[i] <- sum(col3[1:i])
}

#Compute the Ci (col5):
for(i in (1:nrow(A))) {
  col5[i] <- var(r[,1])*col2[i]/(1+var(r[,1])*col4[i])
}

#SHORT SALES ALLOWED:
#Compute the zi:
z_short <- (A[,3]/A[,5])*(A[,6]-col5[nrow(A)])
#Compute the xi:
x_short <- z_short/sum(z_short)

#The final table when short sales allowed:
Weights_with_short <- cbind(A, col1, col2, col3, col4, col5, z_short, x_short)
print(Weights_with_short)
```

```
##       stock        alpha      beta       Ribar         mse       Ratio
##  [1,]    10 1.388844e-02 0.4342799 0.017706285 0.002683414 0.036166273
##  [2,]    17 1.230233e-02 0.4415925 0.016184457 0.001534396 0.032121146
##  [3,]    14 1.420789e-02 0.6063894 0.019538775 0.014905473 0.028923287
##  [4,]     3 3.383866e-02 2.0248862 0.051639839 0.011146357 0.024514878
##  [5,]     6 1.609522e-02 0.9442843 0.024396608 0.005579442 0.023718078
##  [6,]    20 7.126095e-03 0.3437258 0.010147857 0.001435659 0.023704525
##  [7,]     2 1.667838e-02 1.1593250 0.026870239 0.001906601 0.021452345
##  [8,]    28 1.433911e-02 1.1254699 0.024233340 0.022451618 0.019754717
##  [9,]    19 5.838137e-03 0.3725548 0.009113340 0.002604566 0.019093408
## [10,]    13 1.750136e-02 1.6022329 0.031586905 0.003738721 0.018466046
## [11,]     4 1.181871e-02 1.0718453 0.021241516 0.003111891 0.017951765
## [12,]    11 5.867074e-03 0.4776197 0.010065923 0.002165591 0.016887753
## [13,]     8 1.012012e-02 1.0530930 0.019378066 0.003794271 0.016501929
## [14,]    23 8.091456e-03 0.8674676 0.015717535 0.001740458 0.015813312
## [15,]    21 4.799198e-03 0.4144253 0.008442493 0.001099256 0.015545609
## [16,]     5 9.787943e-03 1.2582033 0.020849058 0.002379235 0.014980932
## [17,]     7 8.128193e-03 1.0393931 0.017265705 0.002303044 0.014687132
## [18,]     1 8.655393e-03 1.2929549 0.020022016 0.003900169 0.013938627
## [19,]    22 4.797339e-03 0.5458005 0.009595579 0.001294483 0.013916404
## [20,]    15 6.910948e-03 1.0001717 0.015703656 0.001438093 0.013701304
## [21,]    26 5.679208e-03 0.7875896 0.012603065 0.001402875 0.013462678
## [22,]    27 5.579035e-03 0.9919519 0.014299481 0.006269398 0.012399272
```

```
## [23,]    29  4.663745e-03 0.8974680 0.012553564 0.001263365 0.011759265
## [24,]    30  4.477371e-03 0.9648233 0.012959325 0.005862893 0.011358893
## [25,]    25  3.180270e-03 0.9214674 0.011281073 0.006992219 0.010072058
## [26,]    12  3.144260e-03 1.0354303 0.012246933 0.002236589 0.009896304
## [27,]     9  1.934431e-03 1.0025766 0.010748281 0.002566266 0.008725798
## [28,]    16  8.588360e-04 2.2070856 0.020261764 0.006017150 0.008274153
## [29,]    24  3.196362e-05 0.5747603 0.005084795 0.002977672 0.005367099
## [30,]    18 -3.223398e-03 0.7638836 0.003492054 0.001285056 0.001953248
##             col1       col2      col3       col4        col5     z_short
##  [1,]  2.5418822   2.541882  70.28322   70.28322 0.002786044  3.67880990
##  [2,]  4.0822262   6.624108 127.08843  197.37166 0.006372699  5.37781783
##  [3,]  0.7135183   7.337627  24.66934  222.04099 0.006895485  0.63010311
##  [4,]  9.0177464  16.355373 367.84789  589.88889 0.011421603  2.01282489
##  [5,]  3.7904802  20.145853 159.81397  749.70286 0.012656161  1.74035841
##  [6,]  1.9507615  22.096615  82.29490  831.99776 0.013199281  2.45875206
##  [7,] 15.1225588  37.219174 704.93733 1536.93508 0.015644786  4.87506629
##  [8,]  1.1145280  38.333701  56.41832 1593.35340 0.015739995  0.31680336
##  [9,]  1.0174857  39.351187  53.28989 1646.64329 0.015811800  0.80938574
## [10,] 12.6795005  52.030688 686.63863 2333.28193 0.016385753  2.15609681
## [11,]  6.6274593  58.658147 369.18149 2702.46341 0.016548861  1.55576331
## [12,]  1.7789341  60.437081 105.33870 2807.80212 0.016558642  0.76152130
## [13,]  4.8232503  65.260331 292.28404 3100.08616 0.016554437  0.85124414
## [14,]  6.8370035  72.097335 432.35747 3532.44362 0.016481188  1.18542509
## [15,]  2.4288541  74.526189 156.24053 3688.68416 0.016448925  0.79574286
## [16,]  9.9678867  84.494076 665.37159 4354.05574 0.016260946  0.81757560
## [17,]  6.8896059  91.383682 469.09130 4823.14704 0.016130632  0.56514170
## [18,]  5.9745241  97.358206 428.63074 5251.77778 0.015976450  0.16698706
## [19,]  3.2025689 100.560775 230.12906 5481.90684 0.015901485  0.20301310
## [20,]  9.5306816 110.091456 695.60400 6177.51084 0.015683459  0.18526946
## [21,]  5.9526805 116.044137 442.16170 6619.67254 0.015551862  0.01558655
## [22,]  1.9460392 117.990176 156.94785 6776.62040 0.015486918 -0.16386059
## [23,]  7.4970299 125.487206 637.54238 7414.16278 0.015199070 -1.19034613
## [24,]  1.8035144 127.290720 158.77554 7572.93832 0.015126613 -0.34163919
## [25,]  1.2231032 128.513823 121.43529 7694.37361 0.015054710 -0.44317304
## [26,]  4.7438232 133.257647 479.35301 8173.72662 0.014780447 -1.63820168
## [27,]  3.4177369 136.675384 391.68186 8565.40848 0.014528361 -1.83973526
## [28,]  6.6984001 143.373784 809.55719 9374.96567 0.014032803 -1.89296310
## [29,]  0.5954376 143.969221 110.94217 9485.90785 0.013939717 -1.55727701
## [30,]  0.8869307 144.856152 454.07995 9939.98779 0.013434915 -6.82511681
##           x_short
##  [1,]  0.240965202
##  [2,]  0.352251678
##  [3,]  0.041272294
##  [4,]  0.131841756
##  [5,]  0.113994968
##  [6,]  0.161050368
##  [7,]  0.319321021
##  [8,]  0.020750892
##  [9,]  0.053015460
## [10,]  0.141226189
## [11,]  0.101903830
## [12,]  0.049880298
## [13,]  0.055757221
## [14,]  0.077646359
```

```
## [15,]  0.052121839
## [16,]  0.053551902
## [17,]  0.037017266
## [18,]  0.010937796
## [19,]  0.013297532
## [20,]  0.012135308
## [21,]  0.001020932
## [22,] -0.010733009
## [23,] -0.077968692
## [24,] -0.022377660
## [25,] -0.029028214
## [26,] -0.107303614
## [27,] -0.120504236
## [28,] -0.123990706
## [29,] -0.102002979
## [30,] -0.447051001
```

```
#SHORT SALES NOT ALLOWED:
#First create a matrix up to the maximum of col5:
table1 <- cbind(A, col1, col2, col3, col4, col5)
table2 <- table1[1:which(col5==max(col5)), ]

#Compute the zi:
z_no_short <- (table2[,3]/table2[,5])*(table2[,6]-max(col5))

#Compute the xi:
x_no_short <- z_no_short/sum(z_no_short)

#The final table when short sales are not allowed:
Weights_no_short <- cbind(table2, z_no_short, x_no_short)
print(Weights_no_short)
```

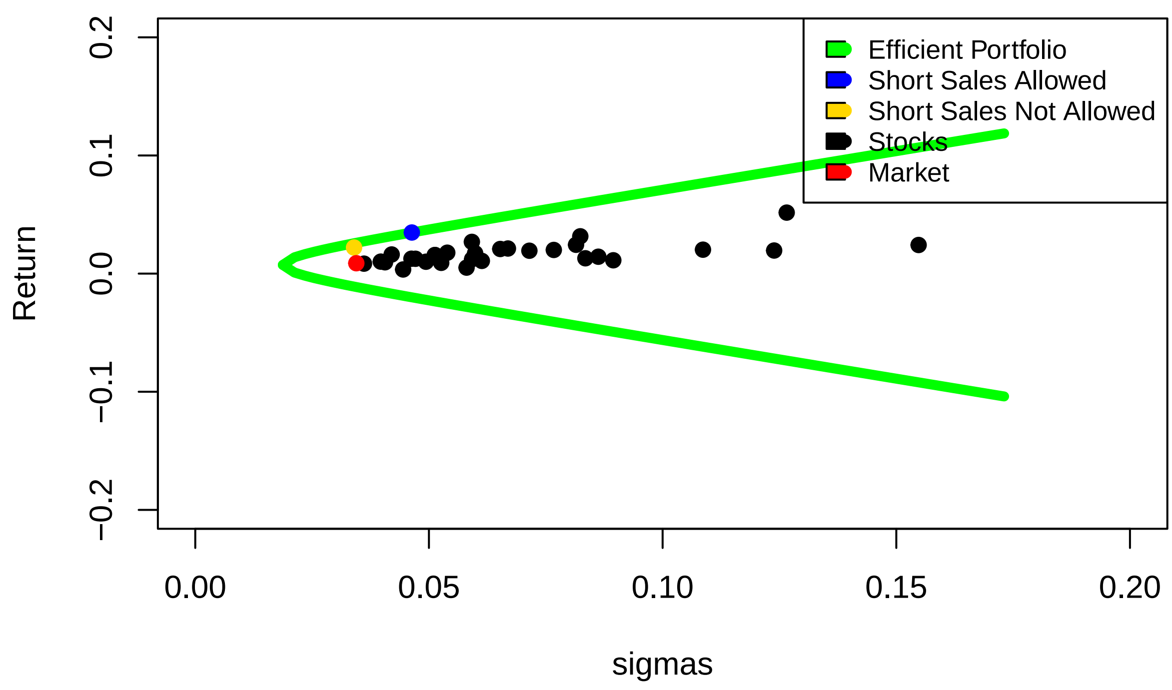
```
##       stock       alpha      beta      Ribar         mse      Ratio       col1
##   [1,]    10 0.013888444 0.4342799 0.01770628 0.002683414 0.03616627  2.5418822
##   [2,]    17 0.012302330 0.4415925 0.01618446 0.001534396 0.03212115  4.0822262
##   [3,]    14 0.014207885 0.6063894 0.01953878 0.014905473 0.02892329  0.7135183
##   [4,]     3 0.033838662 2.0248862 0.05163984 0.011146357 0.02451488  9.0177464
##   [5,]     6 0.016095218 0.9442843 0.02439661 0.005579442 0.02371808  3.7904802
##   [6,]    20 0.007126095 0.3437258 0.01014786 0.001435659 0.02370453  1.9507615
##   [7,]     2 0.016678383 1.1593250 0.02687024 0.001906601 0.02145235 15.1225588
##   [8,]    28 0.014339111 1.1254699 0.02423334 0.022451618 0.01975472  1.1145280
##   [9,]    19 0.005838137 0.3725548 0.00911334 0.002604566 0.01909341  1.0174857
## [10,]    13 0.017501358 1.6022329 0.03158691 0.003738721 0.01846605 12.6795005
## [11,]     4 0.011818710 1.0718453 0.02124152 0.003111891 0.01795177  6.6274593
## [12,]    11 0.005867074 0.4776197 0.01006592 0.002165591 0.01688775  1.7789341
##             col2      col3       col4        col5 z_no_short  x_no_short
##   [1,]  2.541882  70.28322   70.28322 0.002786044 3.17327045 0.182462891
##   [2,]  6.624108 127.08843  197.37166 0.006372699 4.47882246 0.257532066
##   [3,]  7.337627  24.66934  222.04099 0.006895485 0.50302260 0.028923774
##   [4,] 16.355373 367.84789  589.88889 0.011421603 1.44535770 0.083107995
##   [5,] 20.145853 159.81397  749.70286 0.012656161 1.21168792 0.069671994
##   [6,] 22.096615  82.29490  831.99776 0.013199281 1.71086871 0.098374864
##   [7,] 37.219174 704.93733 1536.93508 0.015644786 2.97565759 0.171100162
##   [8,] 38.333701  56.41832 1593.35340 0.015739995 0.16021505 0.009212357
##   [9,] 39.351187  53.28989 1646.64329 0.015811800 0.36257066 0.020847795
```

```
## [10,] 52.030688 686.63863 2333.28193 0.016385753 0.81742011 0.047001615
## [11,] 58.658147 369.18149 2702.46341 0.016548861 0.47984103 0.027590835
## [12,] 60.437081 105.33870 2807.80212 0.016558642 0.07258531 0.004173652
```

Part b: Find the composition of the point of tangency with and without short sales allowed. Place the two portfolios on the plot with the 30 stocks, S&P 500, and the efficient frontier that you constructed in the previous projects. Your answer for the short sales case should be the same as in project 4, part (a).

```
#find the return of the portfolio with short sales allowed

R_p_short <- Weights_with_short[,13] %*% Weights_with_short[,4]

covariance_matrix = matrix(0,n_stocks,n_stocks)
var_Rm = var(r[,1])
for (i in 1:n_stocks)
{
  for(j in 1:n_stocks){
    if(i == j)
    {
      covariance_matrix[i,j] = Weights_with_short[i,3] * Weights_with_short[i,3] * var_Rm + Weights_with
    }else{
      covariance_matrix[i,j] = Weights_with_short[i,3] * Weights_with_short[j,3] * var_Rm
    }
  }
}

#find the risk of the portfolio with short sales allowed

var_p_short <- Weights_with_short[,13] %*% covariance_matrix %*% Weights_with_short[,13]

#find the return of the portfolio with no short sales allowed
n_long = nrow(Weights_no_short)
R_p_no_short <- Weights_no_short[1:n_long,13] %*% Weights_no_short[1:n_long,4]
#find the risk of the portfolio with no short sales allowed
var_p_no_short <- Weights_no_short[1:n_long,13] %*% covariance_matrix[1:n_long,1:n_long]  %*% Weights_no

#Trace out efficient portfolio
inv_covmat_single_index = solve(covariance_matrix)
ones = rep(1,n_stocks)

A = as.numeric(t(Weights_with_short[,4]) %*% inv_covmat_single_index %*% ones)
B = as.numeric(t(Weights_with_short[,4]) %*% inv_covmat_single_index %*% Weights_with_short[,4])
C = as.numeric(t(ones)  %*% inv_covmat_single_index %*% ones)
D = B*C - A^2

E <- seq(-0.2,0.2,.001)

sigmas <- sqrt(seq(1/C,0.03,.0001))
upper_part <- (A + sqrt(D*(C*sigmas^2 - 1)))*(1/C)
lower_part <- (A - sqrt(D*(C*sigmas^2 - 1)))*(1/C)

plot(sigmas, upper_part, lwd=5,type = "l",col = "green",xlim = c(0, 0.2), ylim= c(-0.2,0.2),ylab = "Retu
lines(sigmas,lower_part, lwd=5,type = "l",col = "green")

points(sqrt(var_p_short),R_p_short, pch=19,lwd=1,col="blue")
```

```
points(sqrt(var_p_no_short),R_p_no_short, pch=19,lwd=1,col="gold")


points(sqrt(diag(covariance_matrix)),Weights_with_short[,4], pch=19,lwd=1,col="black")
points(sqrt(var(r[,1])),mean(r[,1]), pch=19,lwd=1,col="red")

legend("topright",
       legend=c("Efficient Portfolio","Short Sales Allowed", "Short Sales Not Allowed", "Stocks","Marke
       col=c("green","blue","gold","black","red"),
       pch = 19,
       fill =c("green","blue","gold","black","red"),
       cex=0.8)
```



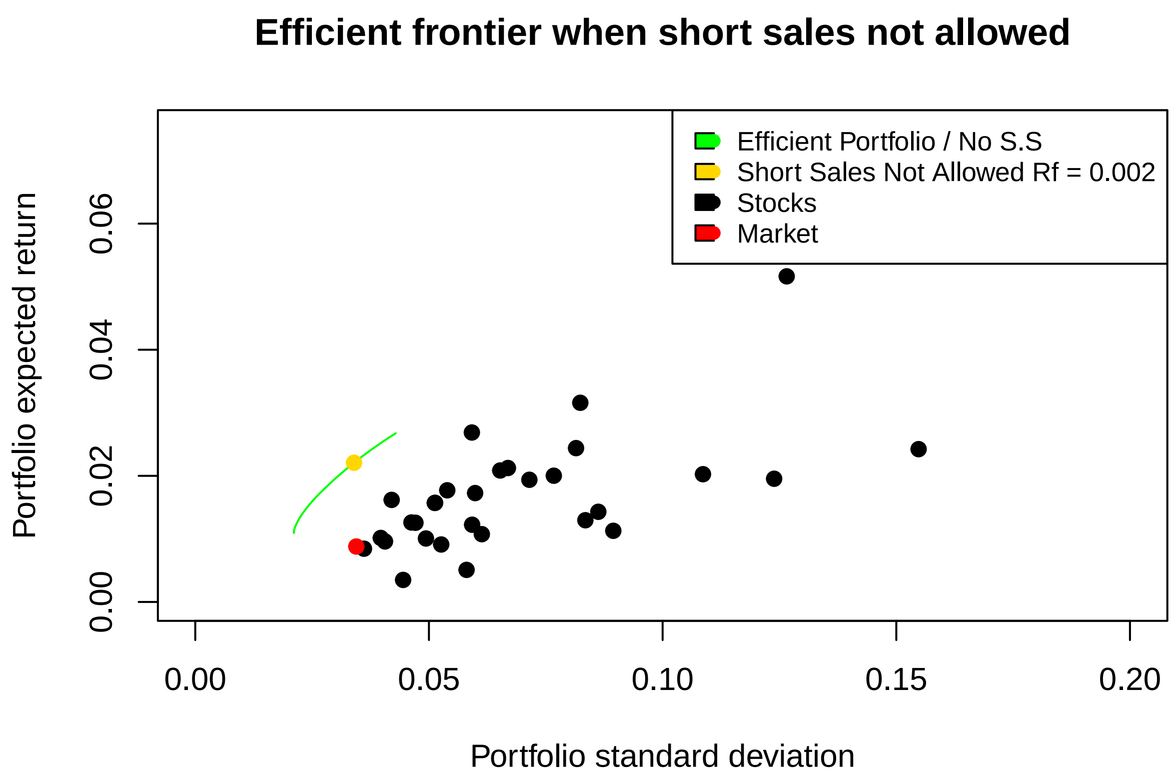
Part c: We want now to draw the efficient frontier when short sale are not allowed. One way to this is to use a for loop where you vary Rf . For each Rf you find the composition of the optimal portfolio (tangency point) and its expected return and standard deviation. Finally connect the points to draw the efficient frontier. Note: See handout #14 under "Labs".

```
Rfr <- seq(-0.2,.007,0.001)

#Initialize the two vectors:
rbar_opt <- rep(0,length(Rfr))
risk_opt <- rep(0,length(Rfr))


for(l in 1:length(Rfr)){
  #Risk free asset:
```

```r
rf <- Rfr[l]
#rf <- .002
#Initialize
beta <- rep(0,ncol(rrr))
alpha <- rep(0,ncol(rrr))
mse <- rep(0,ncol(rrr))
Ribar <- rep(0,ncol(rrr))
Ratio <- rep(0,ncol(rrr))
stocknum <- rep(0,ncol(rrr))
#stock <- names(rrr)

#This for loop computes the required inputs:
for(i in 1:ncol(rrr)){
  q <- lm(data=rrr, formula=rrr[,i] ~ r[,1])
  beta[i] <- q$coefficients[2]
  alpha[i] <- q$coefficients[1]
  mse[i] <- summary(q)$sigma^2
  Ribar[i] <- q$coefficients[1]+q$coefficients[2]*mean(r[,1])
  Ratio[i] <- (Ribar[i]-rf)/beta[i]
  stocknum[i] <- i
}

#So far we have this table:
#xx <- (cbind(stock,alpha, beta, Ribar, mse, Ratio))
xx <- (data.frame(stocknum,alpha, beta, Ribar, mse, Ratio))


#Order the table based on the excess return to beta ratio:
A <- xx[order(-xx[,6]),]

col1 <- rep(0,nrow(A))
col2 <- rep(0,nrow(A))
col3 <- rep(0,nrow(A))
col4 <- rep(0,nrow(A))
col5 <- rep(0,nrow(A))


#Create the last 5 columns of the table:
col1 <- (A[,4]-rf)*A[,3]/A[,5]
col3 <- A[,3]^2/A[,5]
for(i in(1:nrow(A))) {
  col2[i] <- sum(col1[1:i])
  col4[i] <- sum(col3[1:i])
}

#So far we have:
cbind(A, col1, col2, col3, col4)


#Compute the Ci (col5):
for(i in (1:nrow(A))) {
  col5[i] <- var(r[,1])*col2[i]/(1+var(r[,1])*col4[i])
}
```

```r
  #The final table when short sales allowed:
  B <- cbind(A, col1, col2, col3, col4, col5)
  rownames(B) <- NULL

  #SHORT SALES NOT ALLOWED:
  #First create a matrix up to the maximum of col5:
  #table1 <- cbind(A, col1, col2, col3, col4, col5)
  #table2 <- (B[1:which(col5==max(col5)), ], nrow=which(col5==max(col5)), ncol=ncol(B))
  table2 <- B[1:which(col5==max(col5)), ]

  #Compute the Zi:
  z_no_short <- (table2[,3]/table2[,5])*(table2[,6]-max(col5))

  #Compute the xi:
  x_no_short <- z_no_short/sum(z_no_short)

  #Compute the mean and variance for each portfolio when short sales not allowed:
  #First match the columns of the data with the composition of the portfolio:

  r1 <- data.frame(rrr[,table2[,1]])

  beta1 <- rep(0,ncol(r1))
  sigma_e1 <- rep(0,ncol(r1))
  alpha1 <- rep(0,ncol(r1))

  for(i in 1:ncol(r1)){
    q1<- lm(r1[,i] ~ r[,1])
    beta1[i] <- q1$coefficients[2]
    sigma_e1[i] <- summary(q1)$sigma^2
    alpha1[i] <- q1$coefficients[1]
  }

  means1 <- colMeans(r1)
  #means1 <- alpha1 + beta1*mean(r[,1])


  #Construct the variance covariance matrix using SIM:
  xx <- rep(0,ncol(r1)*(ncol(r1)))             #Initialize
  varcovar <- matrix(xx,nrow=ncol(r1),ncol=ncol(r1))  #the variance covariance matrix


  for (i in 1:ncol(r1)){
    for (j in 1:ncol(r1)){
      varcovar[i,j]=beta1[i]*beta1[j]*var(r[,1])
      if(i==j){varcovar[i,j]=beta1[i]^2*var(r[,1])+ sigma_e1[i]}
    }
  }


  rbar_opt[l] <- t(x_no_short) %*% means1
  risk_opt[l] <- ( t(x_no_short) %*% varcovar %*% x_no_short )^.5
}
```

```r
plot(risk_opt, rbar_opt, type="l", main="Efficient frontier when short sales not allowed", ylab="Portfol
     xlim = c(0,0.2),ylim = c(0,0.075))

points(sqrt(var_p_no_short),R_p_no_short, pch=19,lwd=1,col="gold")

points(sqrt(diag(covariance_matrix)),Weights_with_short[,4], pch=19,lwd=1,col="black")
points(sqrt(var(r[,1])),mean(r[,1]), pch=19,lwd=1,col="red")

legend("topright",
       legend=c("Efficient Portfolio / No S.S", "Short Sales Not Allowed Rf = 0.002", "Stocks","Market"
       col=c("green","gold","black","red"),
       pch = 19,
       fill =c("green","gold","black","red"),
       cex=0.8)
```



Part d:Assume the constant correlation model holds. Rank the stocks based on the excess return to standard deviation ratio and complete the entire table based on handout #33: Note: Please use the same Rf as the one in (a) if possible.

```r
#Read all data
a_all <- read.csv("stockData.csv", sep=",", header=TRUE)

# Use 5 year data to train
a <- a_all[1:60,]

#Convert adjusted close prices into returns:
r <- (a[-1,3:ncol(a)]-a[-nrow(a),3:ncol(a)])/a[-nrow(a),3:ncol(a)] # return of stocks + market
```

```r
#r_m <- (a[-1,3]-a[-nrow(a),3])/a[-nrow(a),3] # return of market
rrr <- r[,-1]

n_stocks= ncol(rrr)

#Compute the average correlation:
rho <- (sum(cor(rrr[1:n_stocks]))-n_stocks)/(n_stocks*(n_stocks-1))

#Initialize the vectors:
col1 <- rep(0,n_stocks)
col2 <- rep(0,n_stocks)
col3 <- rep(0,n_stocks)

#Initialize the var-covar matrix:
y <- rep(0,n_stocks*n_stocks)
mat <- matrix(y, ncol=n_stocks, nrow=n_stocks)

#Compute necessary quantities:
R_f = 0.002
Rbar <- colMeans(rrr[1:n_stocks])
Rbar_f <- Rbar-R_f
sigma <- ( diag(var(rrr[1:n_stocks])) )^0.5
Ratio <- Rbar_f/sigma

#Initial table:
xx <- (cbind(Rbar, Rbar_f, sigma, Ratio))

#Order the table based on the excess return to sigma ratio:
aaa <- xx[order(-Ratio),]

#Create the last 3 columns of the table:
for(i in(1:n_stocks)) {

  col1[i] <- rho/(1-rho+i*rho)

  col2[i] <- sum(aaa[,4][1:i])
}

#Compute the Ci:
for(i in (1:n_stocks)) {

  col3[i] <- col1[i]*col2[i]

}

#Create the entire table until now:
xxx <- cbind(aaa, col1, col2, col3)


#SHORT SALES ALLOWED:
#Compute the Zi:
z <- (1/((1-rho)*xxx[,3]))*(xxx[,4]-xxx[,7][nrow(xxx)])
```

```r
#Compute the xi:
x <- z/sum(z)

#The final table:
aaaa <- cbind(xxx, z, x)
print(aaaa)
```

```
##              Rbar      Rbar_f      sigma      Ratio       col1      col2
## MSFT  0.026870239 0.024870239 0.05890517 0.42220812 0.25943203 0.4222081
## NVDA  0.051639839 0.049639839 0.12579059 0.39462282 0.20599129 0.8168309
## AMZN  0.031586905 0.029586905 0.08199295 0.36084693 0.17080662 1.1776779
## MCD   0.016184457 0.014184457 0.04170749 0.34009376 0.14588798 1.5177716
## TMUS  0.017706285 0.015706285 0.05348941 0.29363355 0.12731435 1.8114052
## ASML  0.020849058 0.018849058 0.06494747 0.29022006 0.11293598 2.1016252
## TSM   0.021241516 0.019241516 0.06650219 0.28933657 0.10147572 2.3909618
## AVGO  0.024396608 0.022396608 0.08088349 0.27689963 0.09212706 2.6678614
## COST  0.015717535 0.013717535 0.05103010 0.26881262 0.08435562 2.9366740
## HD    0.015703656 0.013703656 0.05100238 0.26868660 0.07779332 3.2053606
## GOOGL 0.017265705 0.015265705 0.05955063 0.25634833 0.07217833 3.4617090
## META  0.019378066 0.017378066 0.07103405 0.24464416 0.06731933 3.7063531
## AAPL  0.020022016 0.018022016 0.07627691 0.23627091 0.06307328 3.9426240
## LIN   0.012603065 0.010603065 0.04599253 0.23053885 0.05933107 4.1731629
## APD   0.012553564 0.010553564 0.04688371 0.22510085 0.05600806 4.3982637
## PG    0.010147857 0.008147857 0.03938542 0.20687499 0.05303753 4.6051387
## PEP   0.009595579 0.007595579 0.04032281 0.18836929 0.05036623 4.7935080
## KO    0.008442493 0.006442493 0.03583658 0.17977422 0.04795111 4.9732822
## CMCSA 0.012246933 0.010246933 0.05891692 0.17392172 0.04575701 5.1472040
## BABA  0.020261764 0.018261764 0.10815806 0.16884331 0.04375492 5.3160473
## VZ    0.010065923 0.008065923 0.04898117 0.16467397 0.04192068 5.4807212
## VALE  0.024233340 0.022233340 0.15352118 0.14482263 0.04023404 5.6255439
## RIO   0.014299481 0.012299481 0.08561427 0.14366158 0.03867788 5.7692054
## DIS   0.010748281 0.008748281 0.06095641 0.14351700 0.03723760 5.9127224
## TSLA  0.019538775 0.017538775 0.12282162 0.14279877 0.03590075 6.0555212
## WMT   0.009113340 0.007113340 0.05219661 0.13627973 0.03465655 6.1918009
## SCCO  0.012959325 0.010959325 0.08286898 0.13224882 0.03349570 6.3240498
## BHP   0.011281073 0.009281073 0.08876938 0.10455263 0.03241011 6.4286024
## FMX   0.005084795 0.003084795 0.05760759 0.05354842 0.03139267 6.4821508
## TM    0.003492054 0.001492054 0.04422498 0.03373781 0.03043716 6.5158886
##            col3         z          x
## MSFT  0.1095343 5.1321900 0.42710370
## NVDA  0.1682601 2.1071820 0.17536085
## AMZN  0.2011552 2.6765175 0.22274127
## MCD   0.2214246 4.5898773 0.38197213
## TMUS  0.2306179 2.4060153 0.20022993
## ASML  0.2373491 1.9105757 0.15899917
## TSM   0.2426246 1.8479702 0.15378910
## AVGO  0.2457822 1.3117668 0.10916596
## COST  0.2477250 1.8651791 0.15522123
## HD    0.2493556 1.8628562 0.15502792
## GOOGL 0.2498604 1.3156796 0.10949158
## META  0.2495092 0.8804966 0.07327541
## AAPL  0.2486742 0.6717461 0.05590308
## LIN   0.2475982 0.9457761 0.07870801
## APD   0.2463382 0.7711764 0.06417773
```

```
## PG    0.2442452  0.2931278  0.02439426
## PEP   0.2414309 -0.3333983 -0.02774559
## KO    0.2384744 -0.6989957 -0.05817081
## CMCSA 0.2355207 -0.5593016 -0.04654539
## BABA  0.2326032 -0.3680704 -0.03063102
## VZ    0.2297556 -0.9276974 -0.07720349
## VALE  0.2263384 -0.4705883 -0.03916262
## RIO   0.2231406 -0.8621582 -0.07174928
## DIS   0.2201756 -1.2141181 -0.10103958
## TSLA  0.2173977 -0.6104635 -0.05080311
## WMT   0.2145865 -1.6051017 -0.13357745
## SCCO  0.2118285 -1.0766859 -0.08960240
## BHP   0.2083517 -1.4264202 -0.11870748
## FMX   0.2034920 -3.3935492 -0.28241305
## TM    0.1983252 -5.0253219 -0.41821007
```

```
#SHORT SALES NOT ALLOWED:
#Find composition of optimum portfolio when short sales are not allowed:
aaaaa <- aaaa[1:which(aaaa[,7]==max(aaaa[,7])), ]
z_no <- (1/((1-rho)*aaaaa[,3]))*(aaaaa[,4]-aaaaa[,7][nrow(aaaaa)])
x_no <- z_no/sum(z_no)

#Final table:
a_no <- cbind(aaaaa, z_no, x_no)
print(a_no)
```

```
##             Rbar     Rbar_f      sigma     Ratio       col1      col2      col3
## MSFT  0.02687024 0.02487024 0.05890517 0.4222081 0.25943203 0.4222081 0.1095343
## NVDA  0.05163984 0.04963984 0.12579059 0.3946228 0.20599129 0.8168309 0.1682601
## AMZN  0.03158691 0.02958691 0.08199295 0.3608469 0.17080662 1.1776779 0.2011552
## MCD   0.01618446 0.01418446 0.04170749 0.3400938 0.14588798 1.5177716 0.2214246
## TMUS  0.01770628 0.01570628 0.05348941 0.2936335 0.12731435 1.8114052 0.2306179
## ASML  0.02084906 0.01884906 0.06494747 0.2902201 0.11293598 2.1016252 0.2373491
## TSM   0.02124152 0.01924152 0.06650219 0.2893366 0.10147572 2.3909618 0.2426246
## AVGO  0.02439661 0.02239661 0.08088349 0.2768996 0.09212706 2.6678614 0.2457822
## COST  0.01571754 0.01371754 0.05103010 0.2688126 0.08435562 2.9366740 0.2477250
## HD    0.01570366 0.01370366 0.05100238 0.2686866 0.07779332 3.2053606 0.2493556
## GOOGL 0.01726570 0.01526570 0.05955063 0.2563483 0.07217833 3.4617090 0.2498604
##              z         x      z_no       x_no
## MSFT  5.132190 0.4271037 3.9508206 0.27063873
## NVDA  2.107182 0.1753609 1.5539708 0.10644996
## AMZN  2.676518 0.2227413 1.8278012 0.12520786
## MCD   4.589877 0.3819721 2.9213818 0.20012021
## TMUS  2.406015 0.2002299 1.1050333 0.07569688
## ASML  1.910576 0.1589992 0.8391135 0.05748087
## TSM   1.847970 0.1537891 0.8015571 0.05490819
## AVGO  1.311767 0.1091660 0.4514088 0.03092236
## COST  1.865179 0.1552212 0.5014984 0.03435360
## HD    1.862856 0.1550279 0.4984344 0.03414371
## GOOGL 1.315680 0.1094916 0.1471150 0.01007766
```

Part e: Find the composition of the point of tangency with and without short sales allowed. Place the two portfolios on the plot with the 30 stocks, S&P 500, and the efficient frontier that you constructed in the previous projects.

```r
#Var-covar matrix based on the constant correlation model:
for(i in 1:30){

  for(j in 1:30){

    if(i==j){
      mat[i,j]=aaaa[i,3]^2
    } else
    {
      mat[i,j]=rho*aaaa[i,3]*aaaa[j,3]
    }
  }
}


#Calculate the expected return and sd of the point of tangency
#when short sales allowed
sd_p_opt <- (t(x) %*% mat %*% x)^.5
R_p_opt <- t(x) %*% aaaa[,1]


#Calculate the expected return and sd of the point of tangency
#when short sales are not allowed
sd_p_opt_no <- (t(x_no) %*% mat[1:which(aaaa[,7]==max(aaaa[,7])),1:which(aaaa[,7]==max(aaaa[,7]))] %*% x
R_p_opt_no <- t(x_no) %*% aaaaa[,1]

#Trace out efficient portfolio
inv_covmat_const_corr= solve(mat)
ones = rep(1,n_stocks)

A = as.numeric(t(aaaa[,1]) %*% inv_covmat_const_corr %*% ones)
B = as.numeric(t(aaaa[,1]) %*% inv_covmat_const_corr %*% aaaa[,1])
C = as.numeric(t(ones)  %*% inv_covmat_const_corr %*% ones)
D = B*C - A^2

E <- seq(-0.2,0.2,.001)

sigmas <- sqrt(seq(1/C,0.03,.0001))
upper_part <- (A + sqrt(D*(C*sigmas^2 - 1)))*(1/C)
lower_part <- (A - sqrt(D*(C*sigmas^2 - 1)))*(1/C)

plot(sigmas, upper_part, lwd=5,type = "l",col = "green",xlim = c(0, 0.2), ylim= c(-0.2,0.2),ylab = "Retu
lines(sigmas,lower_part, lwd=5,type = "l",col = "green")

points(sd_p_opt,R_p_opt, pch=19,lwd=1,col="blue")
points(sd_p_opt_no,R_p_opt_no, pch=19,lwd=1,col="gold")
points(aaaa[,3],aaaa[,1], pch=19,lwd=1,col="black")
points(sqrt(var(r[,1])),mean(r[,1]), pch=19,lwd=1,col="red")

legend("topright",
       legend=c("Efficient Portfolio const corr","Short Sales Allowed", "Short Sales Not Allowed", "Sto
       col=c("green","blue","gold","black","red"),
       pch = 19,
```

```
    fill =c("green","blue","gold","black","red"),
    cex=0.8)
```