C183 Project Part 6

Maya Lee

5/29/2020

a. Assume the multigroup model holds with short sales allowed. Find the composition of the optimal portfolio and its expected return and standard deviation and place it on the plot you constructed in previous projects with all the other portfolios and stocks.

```
a <- read.csv("stockData.csv", sep=",", header=TRUE)

#Convert adjusted close prices into returns:
r <- (a[-1,3:ncol(a)]-a[-nrow(a),3:ncol(a)])/a[-nrow(a),3:ncol(a)]
sd <- diag(var(r)) ^ 0.5
rf <- 0.005

avgg1 <- mean(cor(r[,2:7]))
avgg2 <- mean(cor(r[,8:13]))
avgg3 <- mean(cor(r[,14:19]))
avgg4 <- mean(cor(r[,20:25]))
avgg5 <- mean(cor(r[,26:31]))

corg12 <- mean(cor(r[,2:7],r[,8:13]))
corg13 <- mean(cor(r[,2:7],r[,14:19]))
corg14 <- mean(cor(r[,2:7],r[,20:25]))
corg15 <- mean(cor(r[,2:7],r[,26:31]))
corg23 <- mean(cor(r[,8:13],r[,14:19]))
corg24<- mean(cor(r[,8:13],r[,20:25]))
corg25<- mean(cor(r[,8:13],r[,26:31]))
corg34<- mean(cor(r[,14:19],r[,20:25]))
corg35<- mean(cor(r[,14:19],r[,26:31]))
corg45<- mean(cor(r[,20:25],r[,26:31]))

cors <- matrix(c(avgg1,corg12,corg13,corg14,corg15,corg12,avgg2,corg23,corg24,corg25,corg13,corg23,avgg3


A <- matrix(data=NA, ncol = 5, nrow = 5)

for (i in 1:5) {
  for (j in 1:5) {
    if (i != j){
      A[i,j] <- (6 * cors[i,j]) / (1-cors[i,i])
    } else {
      A[i,j] <- 1 + (6 * cors[i,j]) / (1-cors[i,i])
    }
  }
}
```

```
C <- matrix(data = NA, ncol =1,nrow = 5)
C[1,1] <- sum((r[,2:7] - rf) / (sd[2:7]*(1 - avgg1)) )
C[2,1] <- sum((r[,8:13] - rf) / (sd[8:13]*(1 - avgg2)) )
C[3,1] <- sum((r[,14:19] - rf) / (sd[14:19]*(1 - avgg3)) )
C[4,1] <- sum((r[,20:25] - rf) / (sd[20:25]*(1 - avgg4)) )
C[5,1] <- sum((r[,26:31] - rf) / (sd[26:31]*(1 - avgg5)) )

phi <- solve(A) %*% C

cstar <- matrix(data = NA, ncol =1,nrow = 5)
cstar[1,1] <- sum(cors[1,] * phi[1])
cstar[2,1] <- sum(cors[2,] * phi[2])
cstar[3,1] <- sum(cors[3,] * phi[3])
cstar[4,1] <- sum(cors[4,] * phi[4])
cstar[5,1] <- sum(cors[5,] * phi[5])

z <- matrix(data = NA, ncol =1,nrow = 30)
for (i in 1:6){
  z[i,1] <- 1/((sd[i+1])*(1-cors[1,1])) * (((mean(r[,i+1])-rf) / sd[i+1]) - cstar[1])
}
for (i in 7:12){
  z[i,1] <- 1/((sd[i+1])*(1-cors[1,1])) * (((mean(r[,i+1])-rf) / sd[i+1]) - cstar[1])
}
for (i in 13:18){
  z[i,1] <- 1/((sd[i+1])*(1-cors[1,1])) * (((mean(r[,i+1])-rf) / sd[i+1]) - cstar[1])
}
for (i in 19:24){
  z[i,1] <- 1/((sd[i+1])*(1-cors[1,1])) * (((mean(r[,i+1])-rf) / sd[i+1]) - cstar[1])
}
for (i in 25:30){
  z[i,1] <- 1/((sd[i+1])*(1-cors[1,1])) * (((mean(r[,i+1])-rf) / sd[i+1]) - cstar[1])
}


x_multigroup <- z/sum(z)
rpmulti <-  as.matrix(r[,-1]) %*% x_multigroup
varmulti <- var(x_multigroup)
sdev <- varmulti^.5

#Convert adjusted close prices into returns:
rr <- (a[-1,3:ncol(a)]-a[-nrow(a),3:ncol(a)])/a[-nrow(a),3:ncol(a)]
rr <- rr[,-1] #remove GSPC

#Compute the means:
means <- colMeans(rr)

#Find the covariance matrix:
cov.matrix <- cov(rr)

ones <- rep(1,30)

#Compute A:
A <- t(ones) %*% solve(cov.matrix) %*% means
```

```r
#Compute B:
B <- t(means) %*% solve(cov.matrix) %*% means


#Compute C:
C <- t(ones) %*% solve(cov.matrix) %*% ones


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


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)

#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)
```

```
## Warning in V * sqrt(D/C): Recycling array of length 1 in vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
```

```
## Warning in A/C + V * sqrt(D/C): Recycling array of length 1 in array-vector arithmetic is deprecated
##   Use c() or as.vector() instead.
```

```r
A2 <- A/C - V * sqrt(D/C)
```

```
## Warning in V * sqrt(D/C): Recycling array of length 1 in vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
```

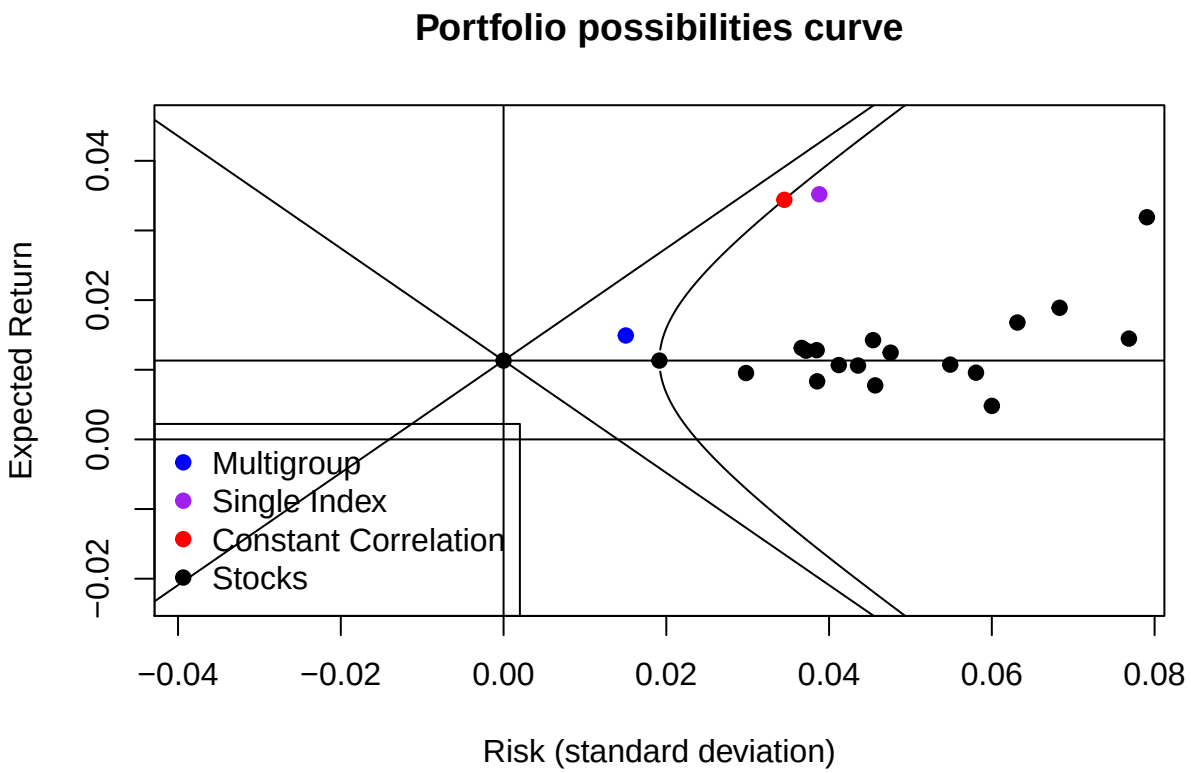
```
## Warning in A/C - V * sqrt(D/C): Recycling array of length 1 in array-vector arithmetic is deprecated
##   Use c() or as.vector() instead.
```

```r
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)
```

```
## Warning in from + (0L:n) * by: Recycling array of length 1 in array-vector arithmetic is deprecated.
##   Use c() or as.vector() instead.
```

```
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(sdev, mean(rpmulti), col = "blue", pch=19)
legend("bottomleft", c("Multigroup", "Single Index", "Constant Correlation", "Stocks"), col = c("blue",
# plot 30 stocks and S&P500
points(apply(r, 2, sd),colMeans(r),pch=19)
# plot SIM from project 5
points(0.0388, 0.0352, col = "purple", pch=19)
# plot constant corr
points(0.0345, 0.0344, col = "red", pch=19)
```

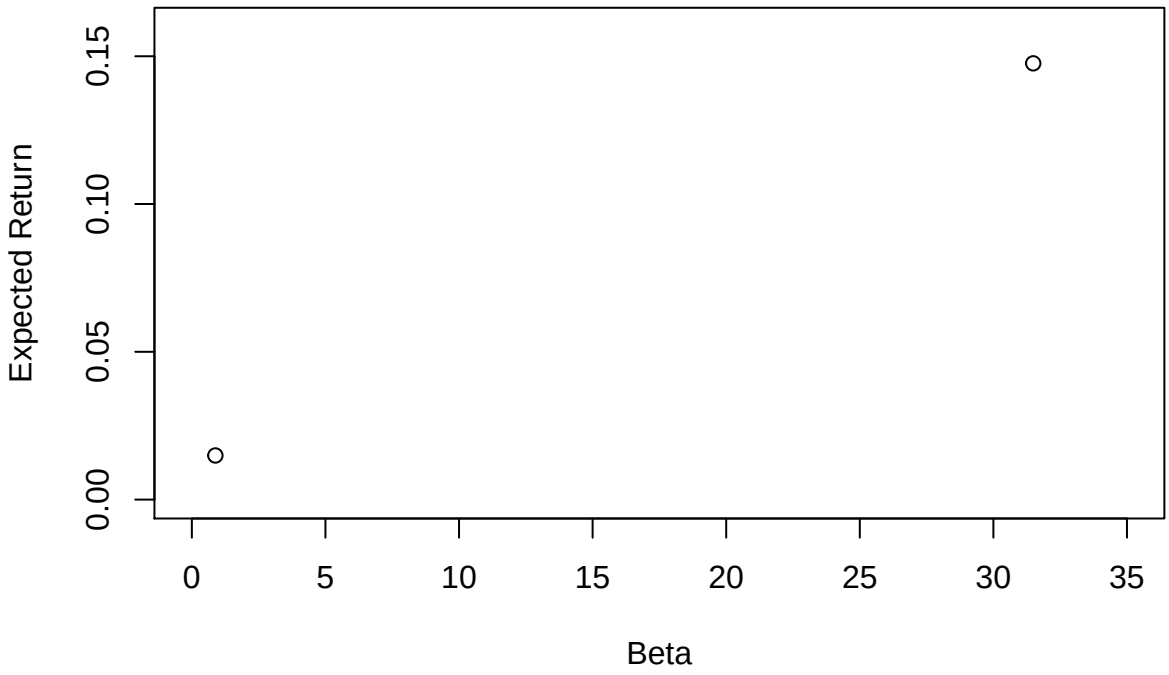


b. Evaluate your portfolios that you constructed in the previous projects. In your analysis you should include the following:

1. Time plots of the performance of all portfolios compared to the S&P500 (see the graph constructed using handout #47).
2. Calculate the Sharpe ratio, differential excess return, Treynor measure, and Jensen differential performance index.
3. Decompose the overall evaluation using Fama's decomposition (net selectivity and diversification) for the single index model when short sales are not allowed. Please show this decomposition on the plot

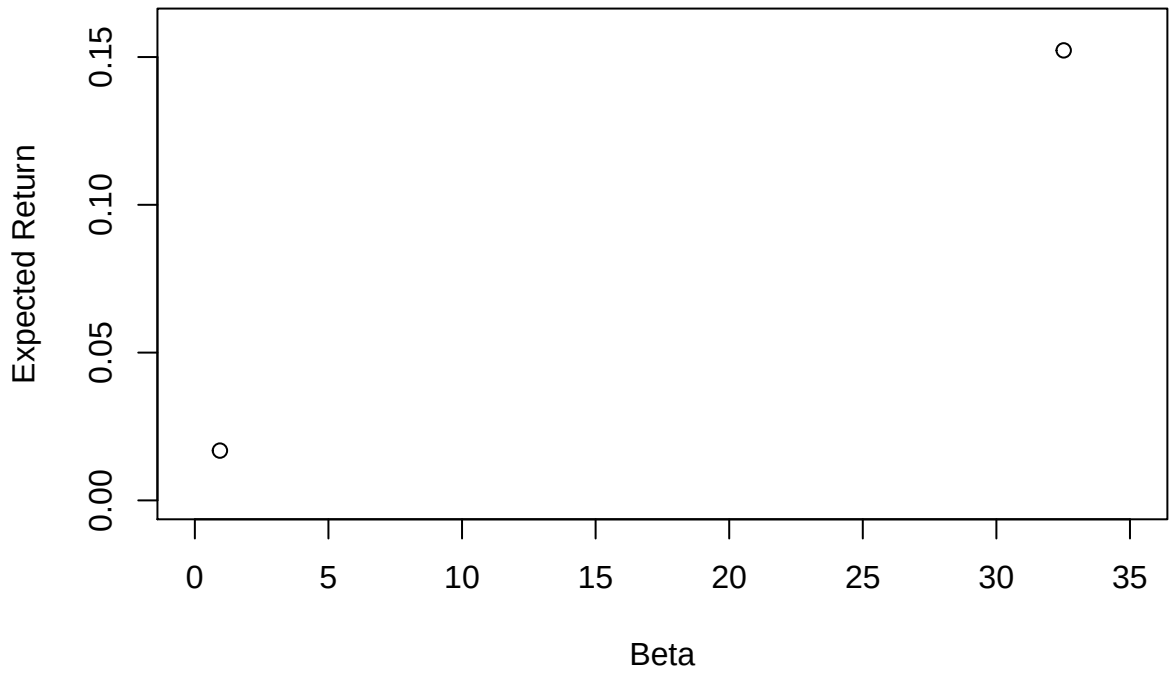
expected return against beta.

```
#Read your csv file:
a <- read.csv("stockData.csv", sep=",", header=TRUE)

#Convert adjusted close prices into returns:
r <- (a[-1,3:ncol(a)]-a[-nrow(a),3:ncol(a)])/a[-nrow(a),3:ncol(a)]


#Initialize
beta <- rep(0,30)

rf <- 0.005

for(i in 1:30){
  q <- lm(data=r, formula=r[,i+1] ~ r[,1])
  beta[i] <- q$coefficients[2]
}

#=====================================================
#Multigroup model
rf <- 0.005
rpmulti <-  as.matrix(r[,-1]) %*% x_multigroup

##### Sharpe
sigmap <- var(rpmulti)^0.5
sharpe <- (rpmulti - rf) %*% sigmap^-1

##### Differential excess return
Rm <- r[,1]
sigmam <- var(Rm)^0.5
Raprime <- rf + (mean(Rm) - rf)/sigmam
dif_excess <- mean(rpmulti) - Raprime

##### Treynor
beta_a <- sum(x_multigroup * beta)
Treynor <- (mean(rpmulti) - rf) / beta_a

##### Jensen
Jensen <- mean(rpmulti) - (rf + beta_a*(mean(Rm) - rf))

##### Net selectivity
beta_2prime <- sqrt(beta_a/var(Rm))
r_2prime <- rf + (mean(Rm)-rf) * beta_2prime
net_selectivity <- mean(rpmulti) - r_2prime

##### Diversification
diversification <- r_2prime - Raprime

plot(beta_2prime, r_2prime, xlim = c(0, 35), ylim = c(0,0.16), xlab = "Beta", ylab = "Expected Return")
points(beta_a, mean(rpmulti))
```



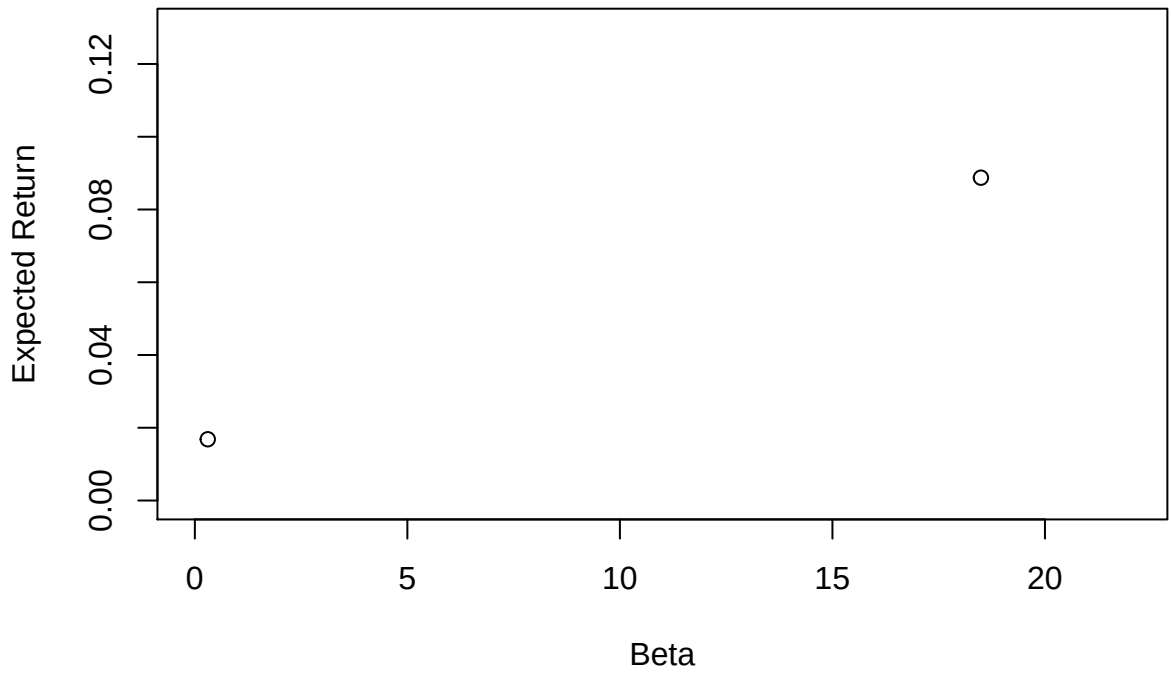

```
#=================================================
#Equal allocation performance for the same time period
aa <- read.csv("stockData.csv", sep=",", header=TRUE)

#Convert adjusted close prices into returns:
rr <- (aa[-1,3:ncol(aa)]-aa[-nrow(aa),3:ncol(aa)])/aa[-nrow(aa),3:ncol(aa)]

#Equal allocation weights:
x <- rep(1/30,30)

rp <-  as.matrix(rr[,-1]) %*% x

##### Sharpe
sigmap <- var(rp)^0.5
sharpe <- (rp - rf) %*% sigmap^-1

##### Differential excess return
Rm <- r[,1]
sigmam <- var(Rm)^0.5
Raprime <- rf + (mean(Rm) - rf)/sigmam
dif_excess <- mean(rp) - Raprime

##### Treynor
beta_a <- sum(x * beta)
Treynor <- (mean(rp) - rf) / beta_a

##### Jensen
Jensen <- mean(rp) - (rf + beta_a*(mean(Rm) - rf))

##### Net selectivity
beta_2prime <- sqrt(beta_a/var(Rm))
r_2prime <- rf + (mean(Rm)-rf) * beta_2prime
```

```r
net_selectivity <- mean(rp) - r_2prime

##### Diversification
diversification <- r_2prime - Raprime

plot(beta_2prime, r_2prime, xlim = c(0, 35), ylim = c(0,0.16), xlab = "Beta", ylab = "Expected Return")
points(beta_a, mean(rp))
```




```r
#=================================================
#Minimum risk

#Use Rf=0.005 to find the optimal portfolio G (tangency point):

#Compute the mean returns:
R_ibar <- as.matrix(colMeans(rr[,-1]))

#Compute the variance-covariance matrix:
var_covar <- cov(rr[,-1])

#Compute the inverse of the variance-covariance matrix:
var_covar_inv <- solve(var_covar)

#Create the vector R:
Rf <- 0.005
R <- R_ibar-Rf

#Compute the vector Z:
z <- var_covar_inv %*% R

#Compute the vector X:
xopt <- z/sum(z)
```

```r
r22 <- as.matrix(rr)

TangencyRet <- r22[,-1] %*% xopt

##### Sharpe
sigmap <- var(TangencyRet)^0.5
sharpe <- (TangencyRet - rf) %*% sigmap^-1

##### Differential excess return
Rm <- r[,1]
sigmam <- var(Rm)^0.5
Raprime <- rf + (mean(Rm) - rf)/sigmam
dif_excess <- mean(TangencyRet) - Raprime

##### Treynor
beta_a <- sum(xopt * beta)
Treynor <- (mean(TangencyRet) - rf) / beta_a

##### Jensen
Jensen <- mean(TangencyRet) - (rf + beta_a*(mean(Rm) - rf))

##### Net selectivity
beta_2prime <- sqrt(beta_a/var(Rm))
r_2prime <- rf + (mean(Rm)-rf) * beta_2prime
net_selectivity <- mean(rp) - r_2prime

##### Diversification
diversification <- r_2prime - Raprime

plot(beta_2prime, r_2prime, xlim = c(0, 22), ylim = c(0,0.13), xlab = "Beta", ylab = "Expected Return")
points(beta_a, mean(rp))
```



```r
#================================================
#Market (S&P500) performance for the period 2016-01-01 to 2019-04-01:
plot(cumprod(1+as.matrix(r)), col="green", lwd=2, type="l", ylim=c(0.7,3),xlim=c(0,70))
points(cumprod(1+rp), col="blue", lwd=2, type="l",lty=2)
lines(cumprod(1+ TangencyRet), col="red", lwd=2,lty=3)
points(cumprod(1+rpmulti), col="purple", lwd=2, type="l",lty=4)


#Add a legend:
legend('bottomright', lty=1:4, c('S&P500','EQUAL','MIN RISK','MULTIGROUP'), col=c("green", "blue","red"
```