Assignment #8 – Efficient Equity Portfolios
Scott Morgan

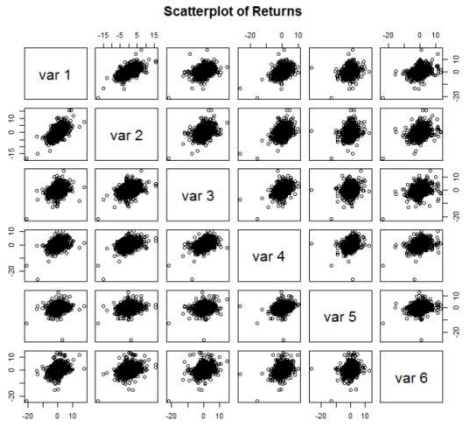
Solve textbook problem 1 at page 489 (Ruppert). (50 pts)

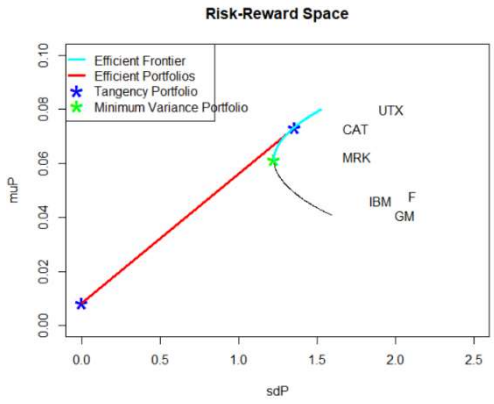
This section uses daily stock prices in the data set Stock Bond.csv that is posted on the book's website and in which any variable whose name ends with "AC" is an adjusted closing price. As the name suggests, these prices have been adjusted for dividends and stock splits, so that returns can be calculated without further adjustments. Run the following code which will read the data, compute the returns for six stocks, create a scatterplot matrix of these returns, and compute the mean vector, covariance matrix, and vector of standard deviations of the returns. Note that returns will be percentages.

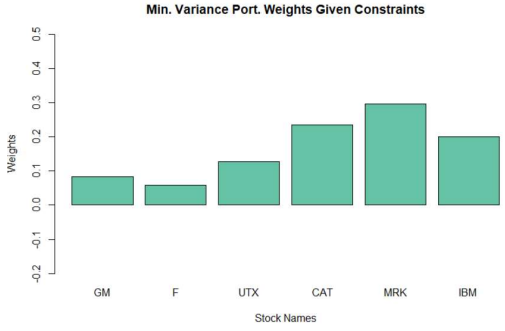
Problem 1 *- Write an* R *program to find the efficient frontier, the tangency portfolio, and the minimum variance portfolio, and plot on "reward-risk space" the location of each of the six stocks, the efficient frontier, the tangency portfolio, and the line of efficient portfolios. Use the constraints that* $-0.1 \leq w_j \leq 0.5$ *for each stock. The first constraint limits short sales but does not rule them out completely. The second constraint prohibits more than 50% of the investment in any single stock. Assume that the annual risk-free rate is 3% and convert this to a daily rate by dividing by 365, since interest is earned on trading as well as non-trading days.*

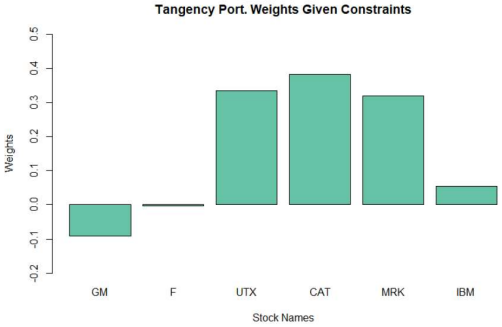
The below code provides the R program to find the efficient frontier, the tangency portfolio, the minimum variance portfolio, and plots on "reward-risk space" the location of each of the six stocks, the efficient frontier, the tangency portfolio, and the line of efficient portfolios.

The minimum and maximum target means used were the minimum and maximum values of the mean vector. Care must be used in this process as too small or too large of values for the target mean are not feasible given the weight constraint. The minimum variance portfolio returns 6.12% with a standard deviation of 1.23% given weights of weights of 0.083, 0.058, 0.128, 0.235, 0.230 and 0.199 for GM, F, UTX, CAT, MRK and IBM, respectively. The Maximum Sharpe Ratio of 0.048 is achieved with weightings of -0.091, -0.003, 0.335, 0.384, 0.320 and 0.056 for GM, F, UTX, CAT, MRK and IBM, respectively. This is also the tangency portfolio which has an expected return of 7.34% and standard deviation of 1.36%.









```
library(MASS)
library(readr)
library(PerformanceAnalytics)
library(RColorBrewer)
# display.brewer.all()
colors <- brewer.pal(n = 6, name = "Set2")

#Set Working Directory-------------------------------------------------------------------------
setwd("C:/Users/SMorgan/Desktop/451/Module8_Current/")

options(scipen=999) #remove scientific notation

Stock_Bond <- read.csv("C:/Users/SMorgan/Desktop/451/Module8_Current/Stock_Bond.csv",header=T)
dat<-Stock_Bond
prices = cbind(dat$GM_AC, dat$F_AC, dat$CAT_AC, dat$UTX_AC,
               dat$MRK_AC, dat$IBM_AC)
n = dim(prices)[1]
returns = 100*(prices[2:n,]/prices[1:(n-1),] - 1)
pairs(returns, main = "Scatterplot of Returns")
mean_vect = apply(returns,2,mean)
cov_mat = cov(returns)
sd_vect = sqrt(diag(cov_mat))

#End of book code
M = length(mean_vect)

#Linear Programming----------------------------------------------------------------------------
library(quadprog)
Amat = cbind(rep(1,M),mean_vect,diag(1,nrow=M),-diag(1,nrow=M))
muP = seq(min(mean_vect), max(mean_vect), length.out=300)
sdP = muP
weights = matrix(0,nrow=300,ncol=M)

# Find the optimal portfolios-------------------------------------------------------------------
for (i in 1:length(muP))
{
  result =
    solve.QP(Dmat=cov_mat,dvec=rep(0,M), Amat=Amat,
             c(1,muP[i],rep(-.1,M),rep(-.5,M)), meq=2) #constraints
  sdP[i] = sqrt(2*result$value)
  weights[i,] = result$solution
}

# Plot efficient frontier and inefficient portfolios below the min var portfolio)------------
plot(sdP,muP,type="l",xlim=c(0,2.5),ylim=c(0,.1),main = "Risk-Reward Space")

# input value of risk-free interest rate
mufree = 3/365
points(0,mufree,cex=3,col="blue",pch="*") # show risk-free asset

#Compute Sharpe's ratios
sharpe =( muP-mufree)/sdP
```

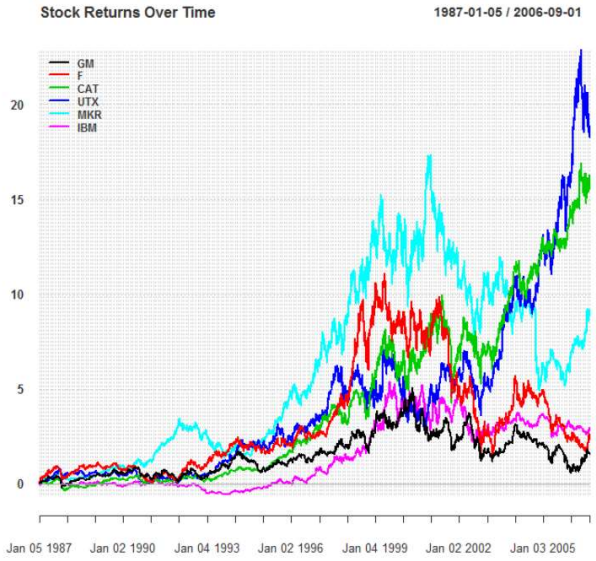
```
#Find Maximum
ind = (sharpe == max(sharpe))

#Line of tangency portfolio
lines(c(0,sdP[ind]),c(mufree,muP[ind]),col="red",lwd=3)
points(sdP[ind],muP[ind],col="blue",cex=3,pch="*")

# min var portfolio
ind2 = (sdP == min(sdP))
points(sdP[ind2],muP[ind2],col="green",cex=3,pch="*")
ind3 = (muP > muP[ind2])

# plot efficient frontier
lines(sdP[ind3],muP[ind3],type="l",xlim=c(0,.25),ylim=c(0,.3),col="cyan",lwd=3)
text(sd_vect[1],mean_vect[1],"GM")
text(sd_vect[2],mean_vect[2],"F")
text(sd_vect[3],mean_vect[3],"UTX")
text(sd_vect[4],mean_vect[4],"CAT")
text(sd_vect[5],mean_vect[5],"MRK")
text(sd_vect[6],mean_vect[6],"IBM")
legend("topleft",c("Efficient Frontier","Efficient Portfolios",
                   "Tangency Portfolio","Minimum Variance Portfolio"),
       lty=c(1,1,NA,NA),
       lwd=c(3,3,1,1),
       pch=c("","","*","*"),
       col=c("cyan","red","blue","green"),
       pt.cex=c(1,1,3,3)
)

# Weights of the minimum variance and tangency portfolio
muP[ind2]
sdP[ind2]
mvWeights=weights[ind2,]
l<-data.frame(mvWeights)
l=t(l)
colnames(l)=c("GM","F","UTX","CAT","MRK","IBM")
l<-as.matrix(l)
barplot(l, xlab = "Stock Names", ylab= "Weights",col=colors,ylim=c(-.2,.5), main="Min. Variance Port. Weights Given
Constraints")

muP[ind]
sdP[ind]
tangWeights=weights[ind,]
q<-data.frame(tangWeights)
q=t(q)
colnames(q)=c("GM","F","UTX","CAT","MRK","IBM")
q<-as.matrix(q)
barplot(q, xlab = "Stock Names", ylab= "Weights",col=colors,ylim=c(-.2,.5), main="Tangency Port. Weights Given
Constraints")
```

PROPOSED EXTRA CREDIT FOR PROBLEM #1 10 POINTS

While the linear programming element of this exercise is interesting, I have spent a considerable amount of time exploring the use of the *PerformanceAnalytics* and *PortfolioAnalytics* packages in R. The combination of these two packages creates attractive visuals, comprehensive return based statistics and robust portfolio simulations

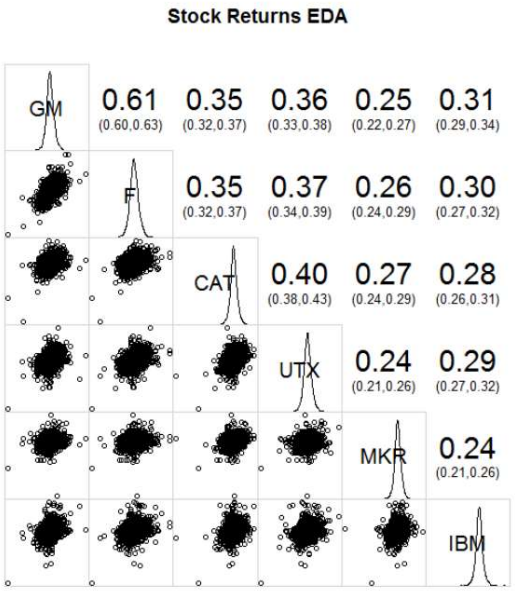
Exploratory Data Analysis

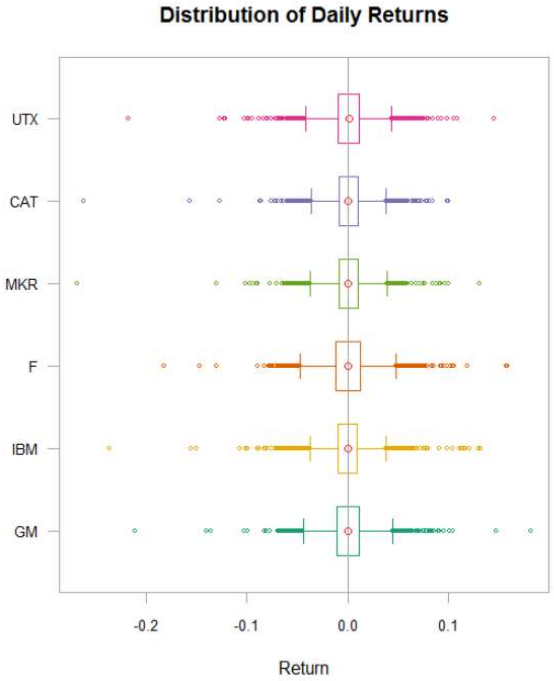
We can see in the first chart, a simple cumulative return of the individual stocks, that the optimizer weighted the 3 top performing companies over the period (UTX, CAT, MRK) while shorting the 2 worst performing stocks (GM, F). *Table.AnnualizedReturns* goes further by illustrating that UTX,CAT and MRK have the highest Sharpe Ratios over the observation period while GM and F have the lowest, corroborating the allocation decisions from the earlier exercise. Using *table.distributions* and *chart.BoxPlot*, we see that the distributions are slightly negative in general while the kurtosis is higher than 3 in all cases, suggesting heavy tails with outliers. Lastly, each stock appears similar in terms of their boxplots with more negative (possible) outliers than positive.



	GM	F	CAT	UTX	MKR	IBM
Annualized Return	0.0506	0.0669	0.1557	0.1640	0.1251	0.0727
Annualized Std Dev	0.3277	0.3345	0.2771	0.3139	0.2790	0.3033
Annualized Sharpe (Rf=2.07%)	0.0887	0.1347	0.4765	0.4466	0.3659	0.1672

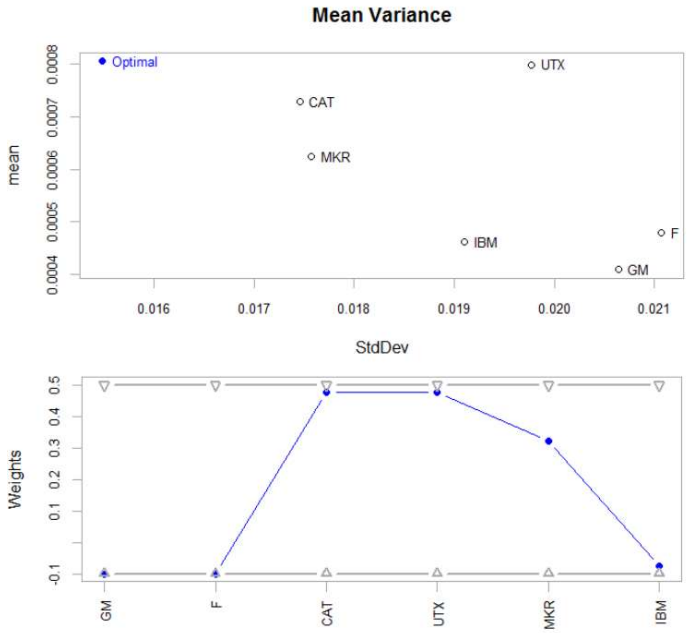
	GM	F	CAT	UTX	MKR	IBM
Monthly Std Dev	0.0206	0.0211	0.0175	0.0198	0.0176	0.0191
Skewness	0.0922	0.1991	-0.7515	-0.1607	-0.7514	-0.1164
Kurtosis	9.0533	7.6748	16.8728	9.4562	17.4171	13.9628
Excess kurtosis	6.0533	4.6748	13.8728	6.4562	14.4171	10.9628
Sample skewness	0.0923	0.1992	-0.7520	-0.1608	-0.7518	-0.1165
Sample excess kurtosis	6.0606	4.6808	13.8880	6.4639	14.4329	10.9750

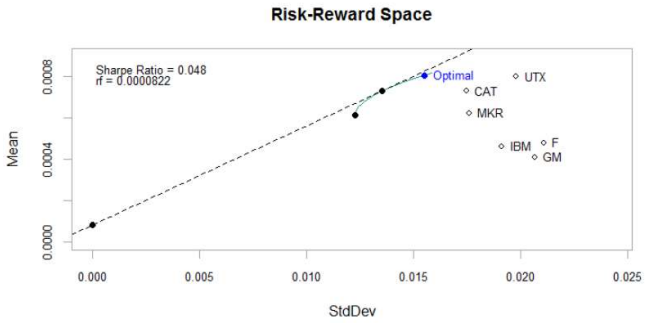




Portfolio Optimization

Shifting to the use of the *PortfolioAnalytics* package, a mean-variance optimization similar to the original exercise is performed with identical constraints. The one exception is that 1000 random portfolios are generated for the efficient frontier. The exhibits on the left plot the risk-reward space and the optimal weights. It appears that this optimizer chooses to short IBM, where the prior solution went long the position. The visuals on the right display all of the same information as the original solution in terms of the risk/return scatter plot. Note the Sharpe Ratio for the tangency portfolio is identical to the submitted response. The bottom right chart shows the return and volatility of the strategy at various weights. Introducing leverage clearly adds more of an opportunity set. Finally, the last two charts show what the optimal weights of the strategy would have been through time assuming no transaction costs and monthly rebalancing. According to R, the strategy cumulative return was 651% during the period versus 601% for the S&P 500 Index (Source: Bloomberg). Therefore, this long/short strategy beats the market over the observation period.





```
library(MASS)
library(readr)
library(PerformanceAnalytics)
library(PortfolioAnalytics)
library(lubridate)
library(xts)
library(tseries)
library(zoo)
library(plotly)
library(quantmod)


setwd("C:/Users/SMorgan/Desktop/451/Module8_Current/")
Stock_Bond <- read.csv("C:/Users/SMorgan/Desktop/451/Module8_Current/Stock_Bond.csv")
mydata<-Stock_Bond
mydata$Date=parse_date_time(x = mydata$Date,
                            orders = c("d m y", "d B Y", "m/d/y"),
                            locale = "eng")
mydata = as.data.frame(mydata[c(1,3, 5, 7, 9, 11,15)])
rownames(mydata) = mydata[,1]
mydata = mydata[, -grep("Date", colnames(mydata))]
mydata = as.xts(mydata)

colnames(mydata) <- c("GM",
                      "F",
                      "CAT",
                      "UTX",
                      "MKR",
                      "IBM")
returns=CalculateReturns(mydata,method="simple")
returns=as.xts(returns)
returns=na.omit(returns)

rf=0.03/365


#Table 1-------------------------------------------------------------------------------
table.AnnualizedReturns(returns,Rf=rf)

#Table 2-------------------------------------------------------------------------------
table.Distributions(returns)

#COLORS##############################################
library(RColorBrewer)
# display.brewer.all()

colors <- brewer.pal(n = 10, name = "Dark2")
themecolor <- "#003366"

############################################
########EDA
############################################
```

```
#Plot 1----------------------------------------------------------------------------------
Return.cumulative = cumprod(1+returns) - 1
chart.TimeSeries(Return.cumulative)
chart.TimeSeries(Return.cumulative,
            legend.loc = "topleft",
            period.areas = cycles.dates,
            period.color = rgb(204/255, 204/255, 204/255, alpha=0.25),lwd=2,
            main = "Stock Returns Over Time")

#Plot 2----------------------------------------------------------------------------------
library(corrgram)
corrgram(as.matrix(returns),
      main="Stock Returns EDA",
      lower.panel=panel.pts, upper.panel=panel.conf,
      diag.panel=panel.density)

#Plot 3----------------------------------------------------------------------------------
chart.Boxplot(returns, main = "Distribution of Daily Returns",colorset = colors)

##############################################
#PORTFOLIO MEAN-VARIANCE OPTIMIZE
##############################################
stocks <- colnames(returns)
init.portf <- portfolio.spec(assets = stocks)

init.portf <- add.constraint(portfolio=init.portf, type="box", min=-0.1, max = 0.5)
meanvar.portf <- add.objective(portfolio=init.portf, type="risk", name="var")
meanvar.portf <- add.objective(portfolio=meanvar.portf, type="return", name="mean")

# Generate random portfolios
set.seed(1234)
rp <- random_portfolios(init.portf, 1000, "sample")

#Optimize
opt_meanvar.portf <- optimize.portfolio(R=returns, portfolio=meanvar.portf, optimize_method="ROI", trace=TRUE)

#Create Frontier
meanvar <- create.EfficientFrontier(R=returns, portfolio=init.portf, type="mean-StdDev")
ef <- extractEfficientFrontier(object=opt_meanvar.portf, match.col="StdDev", n.portfolios=1000)

#Plot 4----------------------------------------------------------------------------------
plot(opt_meanvar.portf, main="Mean Variance", risk.col="StdDev", neighbors=100, chart.assets = TRUE)

#Plot 5----------------------------------------------------------------------------------
par(mfrow=c(2,1))
chart.EfficientFrontier(opt_meanvar.portf
                  , match.col="StdDev"
                  , type="l"
                  , RAR.text="Sharpe Ratio"
                  , main = "Risk-Reward Space"
                  , col = colors
                  , rf = rf)
```

```
chart.EF.Weights(opt_meanvar.portf, match.col="StdDev", colorset=colors, main = "Returns and Std Dev. at Various Weights")
par(mfrow=c(1,1))


#Plot 6-------------------------------------------------------------------------------
par(mfrow=c(1,1))
opt_rebal<- optimize.portfolio.rebalancing(R = returns, portfolio = init.portf, optimize_method = "DEoptim", rp = 50, trace = TRUE, search_size = 1000, rebalance_on = "quarters", training_period = 60, rolling_window = 60)
weightRange<-extractWeights(opt_rebal)
chart.Weights(opt_rebal, col=colors, main= "Optimal Weights")
rr1<-Return.portfolio(returns, weights = extractWeights(opt_rebal))

#Plot 7-------------------------------------------------------------------------------
chart.CumReturns(rr1)
Return.cumulative(rr1)
```