Portfolio Analysis

Xiner Zhou

October 13, 2016

**getPortfolio(er, cov.mat, weights)**

* inputs:
  + er: expected returns vector of chosen stocks
  + cov.mat: covariance matrix of returns of chosen stocks
  + weights: designated portfolio weights of chosen stocks
* outputs:
  + er: portfolio expected return
  + sd: portfolio standard deviation
  + weights: designated portfolio weights of chosen stocks

**efficient.portfolio(er, cov.mat, target.return)**

* inputs:
  + er: expected returns vector of chosen stocks
  + cov.mat: covariance matrix of returns of chosen stocks
  + target.return: target expected return (the function finds the minimum risk portfolio with this target return)
* outputs:
  + er: portfolio expected return
  + sd: portfolio standard deviation
  + weights: portfolio weights of chosen stocks that minimizes the risk/sd at the level of target return

**globalMin.portfolio(er, cov.mat)**

* inputs:
  + er: expected returns vector of chosen stocks
  + cov.mat: covariance matrix of returns of chosen stocks
* outputs:
  + er: GMV portfolio expected return
  + sd: GMV portfolio standard deviation
  + weights: portfolio weights of chosen stocks that minimizes the risk/sd for any level of return

**tangency.portfolio(er, cov.mat, risk.free)**

* inputs:
  + er: expected returns vector of chosen stocks
  + cov.mat: covariance matrix of returns of chosen stocks
  + risk.free: the return of the risk-free asset
* outputs:
  + er: tangency/market portfolio expected return
  + sd: tangency/market portfolio standard deviation
  + weights: tangency/market portfolio weights

**efficient.frontier(er, cov.mat, nport, alpha.min=-0.5, alpha.max=1.5)**

* inputs:
  + er: expected returns vector of chosen stocks
  + cov.mat: covariance matrix of returns of chosen stocks
  + nport: number of efficient portfolios to compute
* outputs:
  + er: nport x 1 vector of expected returns of efficient portfolios
  + sd: nport x 1 vector of std deviations of efficient portfolios
  + weights: nport x N matrix of weights of efficient portfolios

**plot.portfolio(object)**

* plot a portfolio weights

**plot.Markowitz(object)**

* plot efficient frontier curve

# Loading required libraries  
library(quantmod)  
  
# include source code  
source("E:/Course Work at Harvard/Introduction to Financial Statisitcs/Eric Zivot.R")  
  
# Pulling Stock Returns  
getSymbols("AAPL", from="2013-01-01")

## [1] "AAPL"

getSymbols("SNA", from="2013-01-01")

## [1] "SNA"

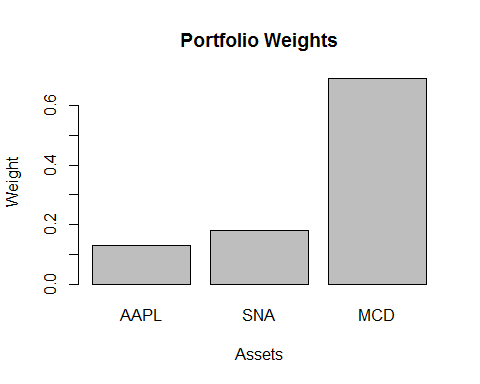
getSymbols("MCD", from="2013-01-01")

## [1] "MCD"

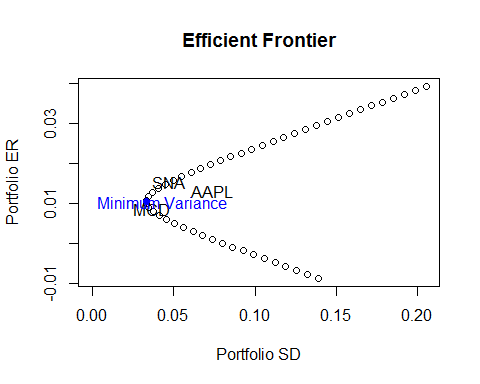
aaplret<-monthlyReturn(Ad(AAPL))  
snaret<-monthlyReturn(Ad(SNA))  
mcdret<-monthlyReturn(Ad(MCD))  
  
# expected return vector  
er<-c(mean(aaplret), mean(snaret), mean(mcdret))  
  
# covariance matrix  
cov.mat<-cov(cbind(aaplret, snaret, mcdret))  
names(er)<-c("AAPL","SNA", "MCD")  
colnames(cov.mat)<-c("AAPL","SNA","MCD")  
rownames(cov.mat)<-c("AAPL","SNA","MCD")  
   
############################################  
## Global Minimum Variance Portfolio  
###########################################  
gmin.port<-globalMin.portfolio(er,cov.mat)  
print(gmin.port)

## Call:  
## globalMin.portfolio(er = er, cov.mat = cov.mat)  
##   
## Portfolio expected return: 0.01048557   
## Portfolio standard deviation: 0.03314065   
## Portfolio weights:  
## AAPL SNA MCD   
## 0.1284 0.1806 0.6911

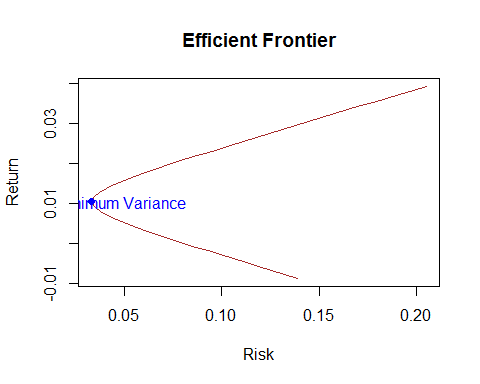
plot(gmin.port)



###########################################  
## Efficient Frontier Curve  
###########################################  
ef<-efficient.frontier(er, cov.mat, nport=50, alpha.min=-5, alpha.max=5)  
  
# Let's plot the efficient frontier using Zivot function  
## Interesting in R, plot.Markowitz overload plot, if the object is a portfolio, R will plot using plot.Markowitz function  
plot(ef, plot.assets=T)  
   
# More interestingly, add a point for the Global Minimum Variance Portfolio  
points(gmin.port$sd, gmin.port$er, col="blue", pch=21, bg="blue")  
text(gmin.port$sd+0.01, gmin.port$er, "Minimum Variance", col="blue")



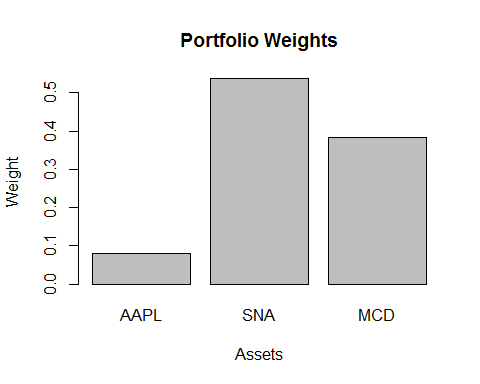
# You can also customize EF  
plot(ef$sd, ef$er,  
 type="l",  
 col="brown",  
 xlab="Risk",  
 ylab="Return",  
 main="Efficient Frontier")  
points(gmin.port$sd, gmin.port$er, col="blue", pch=21, bg="blue")  
text(gmin.port$sd+0.015, gmin.port$er, "Minimum Variance", col="blue")



###########################################  
## Tangency Portfolio and Capital Market Line  
###########################################  
  
# Compute tangency portfolio  
rk.free<-0  
tan.port<-tangency.portfolio(er, cov.mat, rk.free)  
print(tan.port)

## Call:  
## tangency.portfolio(er = er, cov.mat = cov.mat, risk.free = rk.free)  
##   
## Portfolio expected return: 0.01260822   
## Portfolio standard deviation: 0.03634057   
## Portfolio weights:  
## AAPL SNA MCD   
## 0.0799 0.5372 0.3829

plot(tan.port)



# compute slope of tangent line (aka capital market line)  
sr.tan<-(tan.port$er-rk.free)/tan.port$sd  
  
# Let's plot our calculation so we can visualize it  
## first plot the canvas then add on assets  
plot(ef, plot.assets=T)  
  
# Adds points to the plot representing GMV and tangent portfolios  
points(gmin.port$sd, gmin.port$er, col="blue", pch=21, bg="blue")  
points(tan.port$sd, tan.port$er, col="red", pch=21, bg="red")  
  
# Adds a line to the plot representing the CML  
abline(a=rk.free, b=sr.tan,col="red")

