Econ 424 Lab 8, Winter 2020

Code

Introduction

In this lab, you will estimate the CER model for the monthly return data on five Northwest stocks in the **IntroCompFin** package: Amazon (amzn), Boeing (ba), Costco (cost), Nordstrom (jwn), and Starbucks (sbux). Then you will apply mean-variance portfolio theory with matrix algebra to analyze a number of portfolios and to do some asset allocation examples. This notebook walks you through all of the computations for the lab. You will use the following R packages

- **IntroCompFinR**
- **PerformanceAnalytics package**.
- **zoo**
- **xts**

Make sure to install these packages before you load them into R. As in the previous labs, use this notebook to answer all questions. Insert R chunks where needed. I will provide code hints below.

Reading

- Zivot, chapters 12 (portfolio theory with matrix algebra) and 13 (portfolio theory with short sales constraints).

Load packages and set options

Hide

```
library(IntroCompFinR)
library(corrplot)
library(PerformanceAnalytics)
library(xts)
library(boot)
library(quadprog)
options(digits = 3)
Sys.setenv(TZ="UTC")
```

Loading data and computing returns

Load the daily price data from **IntroCompFinR**, and create monthly simple returns over the period Jan 2010 through Dec 2014:

Hide

```
data(amznDailyPrices, baDailyPrices, costDailyPrices, jwnDailyPrices, sbuxDailyPrices)
fiveStocks = merge(amznDailyPrices, baDailyPrices, costDailyPrices, jwnDailyPrices, sbuxDailyPri
ces)
fiveStocks = to.monthly(fiveStocks, OHLC=FALSE)
```

```
missing values removed from data
```

Hide

```
smpl = "2010::2014"
fiveStocksRet = na.omit(Return.calculate(fiveStocks, method = "simple"))
fiveStocksRet = fiveStocksRet[smpl]
```

We will construct portfolios using the 5 years of simple returns from Jan 2010 - Dec 2014.

Hide

```
head(fiveStocksRet, n=3)
```

```
            AMZN     BA    COST     JWN    SBUX
Jan 2010 -0.0677 0.1195 -0.0294 -0.0808 -0.0547
Feb 2010 -0.0559 0.0494  0.0649  0.0740  0.0509
Mar 2010  0.1467 0.1495 -0.0208  0.1058  0.0598
```

Hide

```
tail(fiveStocksRet, n=3)
```

```
            AMZN      BA     COST    JWN    SBUX
Oct 2014 -0.0527 -0.0194  0.06421 0.0620 0.00134
Nov 2014  0.1086  0.0819  0.06835 0.0563 0.07909
Dec 2014 -0.0835 -0.0326 -0.00255 0.0397 0.01038
```

Part I: CER Model Estimation

Consider the CER Model for cc returns

$$
\begin{aligned}
R_{it} &= \mu_i + \epsilon_{it}, t = 1, \cdots, T \\
\epsilon_{it} &\sim \text{iid } N(0, \sigma_i^2) \\
\text{cov}(R_{it}, R_{jt}) &= \sigma_{i,j} \\
\text{cov}(R_{it}, R_{js}) &= 0 \text{ for } s \neq t
\end{aligned}
$$

where R_{it} denotes the cc return on asset i ($i = \text{AMZN}, \cdots, \text{SBUX}$).

1. Using sample descriptive statistics, give estimates for the model parameters $\mu_i, \sigma_i^2, \sigma_i, \sigma_{i,j}, \rho_{i,j}$. Put the estimated mean values in the vector `muhat.vals` and put the estimated covariance matrix in the matrix object `sigma.mat`. These will be inputs for the portfolio theory examples.

Hide

```
muhat = apply(fiveStocksRet, 2, mean)
sigma2hat = apply(fiveStocksRet, 2, var)
sigmahat = apply(fiveStocksRet, 2, sd)
cov.mat = var(fiveStocksRet)
cor.mat = cor(fiveStocksRet)
covhat.vals = cov.mat[lower.tri(cov.mat)]
rhohat.vals = cor.mat[lower.tri(cor.mat)]
muhat.vec = c(muhat)
sigma.mat = cov.mat
sigma.mat
```

```
        AMZN       BA     COST     JWN     SBUX
AMZN 0.00643 0.001665 0.001397 0.00296 0.002318
BA   0.00166 0.003778 0.000625 0.00170 0.001165
COST 0.00140 0.000625 0.001723 0.00157 0.000707
JWN  0.00296 0.001699 0.001566 0.00557 0.001903
SBUX 0.00232 0.001165 0.000707 0.00190 0.003679
```

2. Show the estimated risk-return tradeoff of these assets (i.e., plot the means on the y-axis and the standard deviations on the horizontal axis. Briefly comment.
    - Assuming a risk free rate of 0.005 (0.5% per month or about 6% per year) compute the Sharpe ratios for each asset. Which asset has the highest Sharpe ratio?
    - Using the bootstrap, compute estimated standard errors and 95% confidence intervals for the Sharpe ratios. How well are the Sharpe ratios estimated?

Hide

```
asset.names = names(muhat.vec)
cex.val = .8
r.f = 0.005
cov2cor(sigma.mat)
```

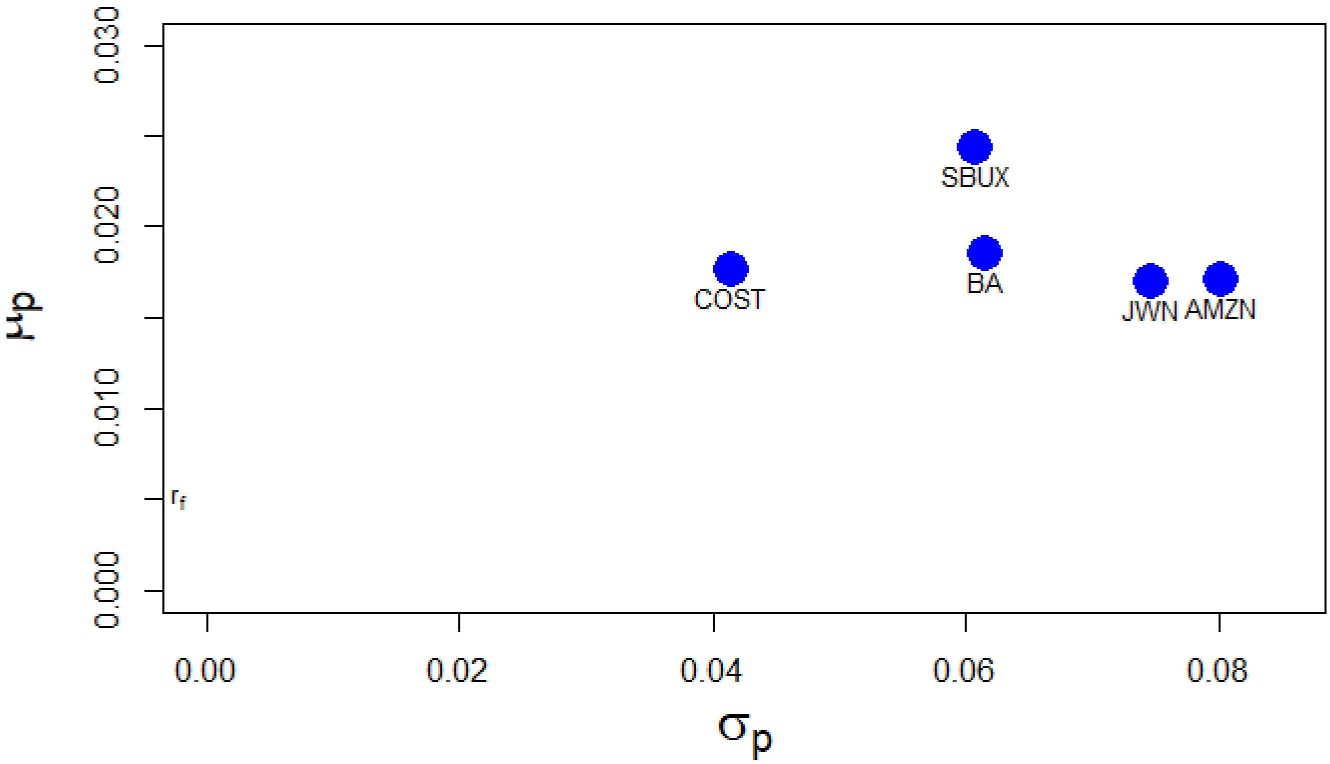
```
      AMZN    BA  COST   JWN  SBUX
AMZN 1.000 0.338 0.420 0.494 0.477
BA   0.338 1.000 0.245 0.370 0.313
COST 0.420 0.245 1.000 0.505 0.281
JWN  0.494 0.370 0.505 1.000 0.420
SBUX 0.477 0.313 0.281 0.420 1.000
```

Hide

```
sd.vec = sqrt(diag(sigma.mat))
plot(sd.vec, muhat.vec,  ylim=c(0, 0.03), xlim=c(0, 0.085), ylab=expression(mu[p]),
     xlab=expression(sigma[p]), pch=16, col="blue", cex=2.5, cex.lab=1.75)
text(sd.vec, muhat.vec, labels=asset.names, pos=1, cex = cex.val)
```

Hide

```
text(0, r.f, labels=expression(r[f]), pos=2, cex = cex.val)
```



Hide

```
sharpe = (muhat.vec - r.f) / sd.vec
sharpe
```

```
 AMZN    BA  COST   JWN  SBUX
0.151 0.220 0.305 0.160 0.320
```

- SBUX has the highest sharpe value.
- AMZN has the lowest sharpe value.

Hide

```
set.seed(123)
sharpeRatio.boot = function(x, idx, rf){
  muhat.boot = mean(x[idx])
  sigmahat.boot = sd(x[idx])
  sharpeRatio = (muhat.boot - rf)/sigmahat.boot
  sharpeRatio
}
sharpe.boot.amzn = boot(fiveStocksRet[,"AMZN"], sharpeRatio.boot, R = 999, rf = r.f)
sharpe.boot.amzn
```

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = fiveStocksRet[, "AMZN"], statistic = sharpeRatio.boot,
    R = 999, rf = r.f)


Bootstrap Statistics :
    original  bias    std. error
t1*    0.151 0.00377       0.127
```

Hide

```
boot.ci(sharpe.boot.amzn, conf = 0.95, type = c("norm", "perc"))
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 999 bootstrap replicates

CALL : 
boot.ci(boot.out = sharpe.boot.amzn, conf = 0.95, type = c("norm", 
    "perc"))

Intervals : 
Level      Normal             Percentile     
95%   (-0.1025,  0.3969 )   (-0.0935,  0.4138 )  
Calculations and Intervals on Original Scale
```

Hide

```
sharpe.boot.ba = boot(fiveStocksRet[,"BA"], sharpeRatio.boot, R = 999, rf = r.f)
sharpe.boot.ba
```

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = fiveStocksRet[, "BA"], statistic = sharpeRatio.boot,
    R = 999, rf = r.f)


Bootstrap Statistics :
    original  bias    std. error
t1*     0.22 0.00489       0.129
```

Hide

```
boot.ci(sharpe.boot.ba, conf = 0.95, type = c("norm", "perc"))
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 999 bootstrap replicates

CALL :
boot.ci(boot.out = sharpe.boot.ba, conf = 0.95, type = c("norm",
    "perc"))

Intervals :
Level      Normal              Percentile
95%   (-0.0386,  0.4682 )   (-0.0167,  0.4782 )
Calculations and Intervals on Original Scale
```

Hide

```
sharpe.boot.cost = boot(fiveStocksRet[,"COST"], sharpeRatio.boot, R = 999, rf = r.f)
sharpe.boot.cost
```

```


ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = fiveStocksRet[, "COST"], statistic = sharpeRatio.boot,
    R = 999, rf = r.f)


Bootstrap Statistics :
    original  bias    std. error
t1*    0.305 -0.0025       0.128
```

Hide

```
boot.ci(sharpe.boot.cost, conf = 0.95, type = c("norm", "perc"))
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 999 bootstrap replicates

CALL :
boot.ci(boot.out = sharpe.boot.cost, conf = 0.95, type = c("norm",
    "perc"))

Intervals :
Level      Normal              Percentile
95%   ( 0.0571,  0.5572 )   ( 0.0531,  0.5569 )
Calculations and Intervals on Original Scale
```

Hide

```
sharpe.boot.jwn = boot(fiveStocksRet[,"JWN"], sharpeRatio.boot, R = 999, rf = r.f)
sharpe.boot.jwn
```

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = fiveStocksRet[, "JWN"], statistic = sharpeRatio.boot,
    R = 999, rf = r.f)


Bootstrap Statistics :
    original  bias    std. error
t1*     0.16 0.00508       0.134
```

Hide

```
boot.ci(sharpe.boot.jwn, conf = 0.95, type = c("norm", "perc"))
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 999 bootstrap replicates

CALL : 
boot.ci(boot.out = sharpe.boot.jwn, conf = 0.95, type = c("norm", 
    "perc"))

Intervals : 
Level      Normal             Percentile     
95%   (-0.1073,  0.4179 )   (-0.0865,  0.4420 )  
Calculations and Intervals on Original Scale
```

Hide

```
sharpe.boot.sbux = boot(fiveStocksRet[,"SBUX"], sharpeRatio.boot, R = 999, rf = r.f)
sharpe.boot.sbux
```

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = fiveStocksRet[, "SBUX"], statistic = sharpeRatio.boot,
    R = 999, rf = r.f)


Bootstrap Statistics :
    original  bias    std. error
t1*     0.32   0.015       0.148
```

Hide

```
boot.ci(sharpe.boot.sbux, conf = 0.95, type = c("norm", "perc"))
```

```
BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
Based on 999 bootstrap replicates

CALL :
boot.ci(boot.out = sharpe.boot.sbux, conf = 0.95, type = c("norm",
    "perc"))

Intervals :
Level      Normal              Percentile
95%   ( 0.0153,  0.5941 )   ( 0.0671,  0.6595 )
Calculations and Intervals on Original Scale
```

- Due to the large range on the intervals I believe that the sharpe ratios are not well estimated.

3. Compute the global minimum variance portfolio allowing short-sales. The minimization problem is

$$\min_{\mathbf{m}} \sigma^2_{p,m} = \mathbf{m}'\Sigma\mathbf{m} \text{ s.t. } \mathbf{m}'\mathbf{1} = 1$$

where $\mathbf{m}$ is the vector of portfolio weights and Σ is the covariance matrix. Briefly comment on the weights. Compute the expected return and standard deviation and add the points to the risk return graph.

Hide

```
#Portfolio weights w/short
one.vec = rep(1, 5)
sigma.inv.mat = solve(sigma.mat)
top.mat = sigma.inv.mat%*%one.vec
bot.val = as.numeric((t(one.vec)%*%sigma.inv.mat%*%one.vec))
m.mat = top.mat/bot.val
m.mat[, 1]
```

```
   AMZN      BA    COST     JWN    SBUX
-0.0491  0.2158  0.6901 -0.0774  0.2207
```

- These weights let us know that at global min, we can short-sale AMZN and JWN as part of our portfolio since they are more risky.

Hide

```
#Expected return and sd
mu.gmin = as.numeric(crossprod(m.mat[, 1], muhat.vec))
sig2.gmin = as.numeric(t(m.mat[, 1])%*%sigma.mat%*%m.mat[, 1])
sig.gmin = sqrt(sig2.gmin)
mu.gmin
```

```
[1] 0.0194
```

Hide

```
sig.gmin
```

```
[1] 0.0359
```

Hide

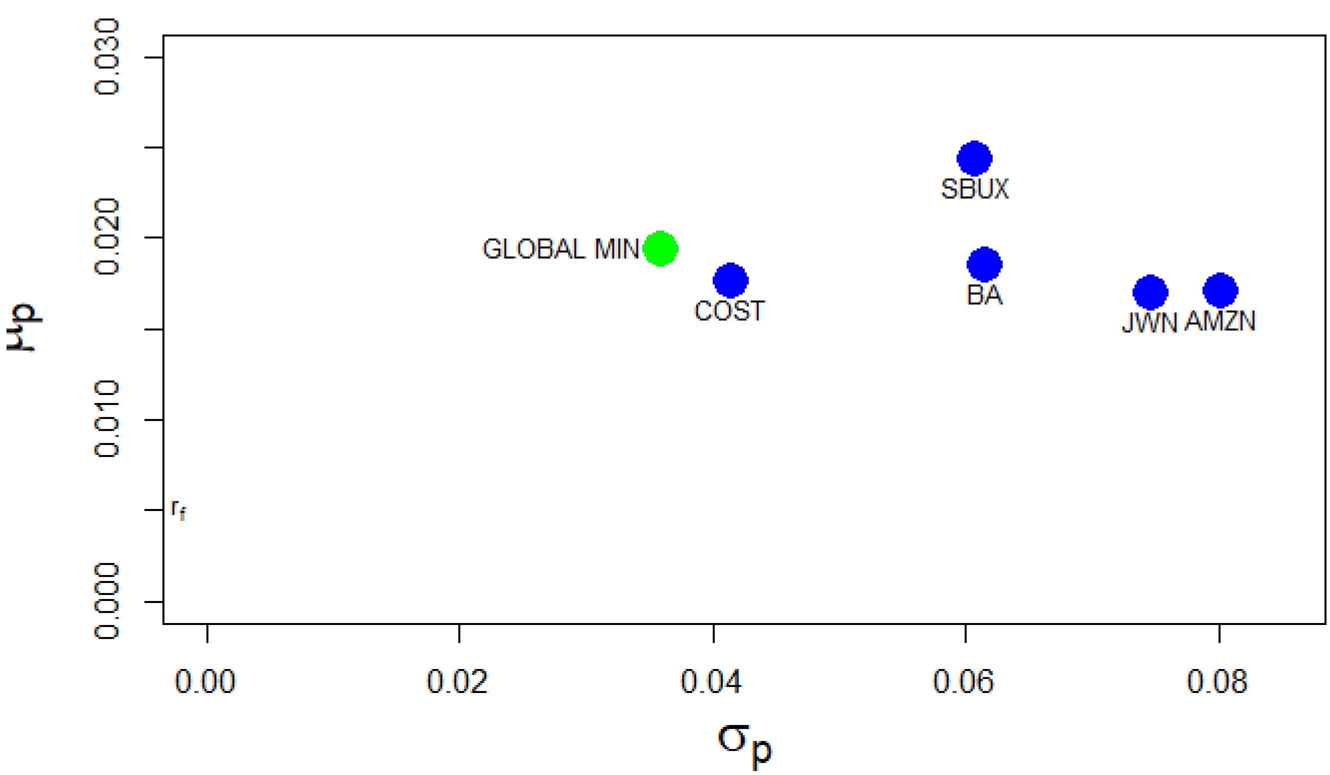
```
#Risk-Return plot
plot(sd.vec, muhat.vec,  ylim=c(0, 0.03), xlim=c(0, 0.085), ylab=expression(mu[p]),
     xlab=expression(sigma[p]), pch=16, col="blue", cex=2.5, cex.lab=1.75)
text(sd.vec, muhat.vec, labels=asset.names, pos=1, cex = cex.val)
```

Hide

```
text(0, r.f, labels=expression(r[f]), pos=2, cex = cex.val)
points(sig.gmin, mu.gmin, pch=16, cex=2.5, col="green")
```

Hide

```
text(sig.gmin, mu.gmin, labels="GLOBAL MIN", pos=2.5, cex = cex.val)
```



4. Of the five stocks, determine the stock with the largest estimated expected return. Use this maximum average return as the target return for the computation of an efficient portfolio allowing for short-sales. That is, find the minimum variance portfolio that has an expected return equal to this target return. The minimization problem is

$$\min_{\mathbf{x}} \sigma^2_{p,x} = \mathbf{x}'\Sigma\mathbf{x} \text{ s.t.}$$
$$\mu_{p,x} = \mathbf{x}'\mu = \mu_p^0 = \text{ target return}$$
$$\mathbf{x}'\mathbf{1} = 1$$

where $\mathbf{x}$ is the vector of portfolio weights, μ is the vector of expected returns and μ_p^0 is the target expected return. Are there any negative weights in this portfolio? Compute the expected return, variance and standard deviation of this portfolio. Finally, compute the covariance between the global minimum variance portfolio and the above efficient portfolio using the formula $\text{cov}(R_{p,m}, R_{p,x}) = \mathbf{m}'\Sigma\mathbf{x}$.

Hide

```
#computing minimum variance portfolio w/ expected return equal to SBUX
top.mat = cbind(2*sigma.mat, muhat.vec, rep(1, 5))
mid.vec = c(muhat.vec, 0, 0)
bot.vec = c(rep(1, 5), 0, 0)
Ax.mat = rbind(top.mat, mid.vec, bot.vec)
bsbux.vec = c(rep(0, 5), muhat.vec["SBUX"], 1)
z.mat = solve(Ax.mat)%*%bsbux.vec
x.vec = z.mat[1:5,]
x.vec
```

```
  AMZN     BA   COST    JWN   SBUX
-0.264  0.134  0.440 -0.247  0.937
```

Hide

```
#Computing expected return and variance of efficient portfolio
mu.px = as.numeric(crossprod(x.vec, muhat.vec))
sig2.px = as.numeric(t(x.vec)%*%sigma.mat%*%x.vec)
sig.px = sqrt(sig2.px)
mu.px
```

```
[1] 0.0244
```

Hide

```
sig.px
```

```
[1] 0.0532
```

Hide

```
#calculating variance for global min and efficient portfolio
sigma.gx = as.numeric(t(m.mat[, 1])%*%sigma.mat%*%x.vec)
rho.gx = sigma.gx/(sig.gmin*sig.px)
sigma.gx
```

```
[1] 0.00129
```

Hide

```
rho.gx
```

```
[1] 0.675
```

5. Repeat question 4 but this time do not allow short sales. That is, add the following constraint: $x_i \geq 0$ for $i = AMZN, \ldots, SBUX$.

Hide

```
D.mat = 2*sigma.mat
d.vec = rep(0, 5)
A.mat = cbind(muhat.vec, rep(1,5), diag(5))
b.vec = c(muhat.vec["SBUX"], 1, rep(0,5))
qp.out = solve.QP(Dmat = D.mat, dvec = d.vec, Amat = A.mat, bvec = b.vec, meq = 2)
names(qp.out$solution) = names(muhat.vec)
weight = round(qp.out$solution, digits=5)
weight
```

```
AMZN   BA COST  JWN SBUX 
   0    0    0    0    1 
```

Hide

```
#Calculating ER and sd for no short efficient portfolio
mu.ns = as.numeric(crossprod(weight, muhat.vec))
sig2.ns = as.numeric(t(weight)%*%sigma.mat%*%weight)
sig.ns = sqrt(sig2.ns)
mu.ns
```

```
[1] 0.0244
```

Hide

```
sig.ns
```

```
[1] 0.0607
```

Hide

```
#Double checking answers
efficient.portfolio(muhat.vec, sigma.mat, target.return = muhat.vec["SBUX"], shorts = FALSE)
```

```
Call:
efficient.portfolio(er = muhat.vec, cov.mat = sigma.mat, target.return = muhat.vec["SBUX"], 
    shorts = FALSE)

Portfolio expected return:     0.0244 
Portfolio standard deviation:  0.0607 
Portfolio weights:
AMZN   BA COST  JWN SBUX 
   0    0    0    0    1 
```

Hide

```
#calculating variance between global min and efficient portfolio w/o short-sales
sigma.ns = as.numeric(t(m.mat[, 1])%*%sigma.mat%*%weight)
rho.ns = sigma.ns/(sig.gmin*sig.ns)
sigma.ns
```

```
[1] 0.00129
```

Hide

```
rho.ns
```

```
[1] 0.592
```

6. Using the fact that all efficient portfolios (that allow short sales) can be written as a convex combination of two efficient portfolios (that allow short sales), compute efficient portfolios as convex combinations of the global minimum variance portfolio and the efficient portfolio computed in question 4. That is, compute

$$\mathbf{z} = \alpha \times \mathbf{m} + (1 - \alpha) \times \mathbf{x}$$

for values of α between 1 and -1 (e.g., make a grid for $\alpha = 1, 0.9, \cdots, -1$). Compute the expected return, variance and standard deviation of these portfolios.

Hide

```
ef = efficient.frontier(muhat.vec, sigma.mat, alpha.min = -1,
                        alpha.max = 1, nport = 20)
ef$weights
```

```
           AMZN     BA  COST     JWN  SBUX
port 1  -0.4791 0.0522 0.190 -0.4169 1.654
port 2  -0.4565 0.0608 0.217 -0.3990 1.578
port 3  -0.4338 0.0694 0.243 -0.3812 1.503
port 4  -0.4112 0.0780 0.269 -0.3633 1.427
port 5  -0.3886 0.0866 0.296 -0.3454 1.352
port 6  -0.3660 0.0952 0.322 -0.3276 1.276
port 7  -0.3433 0.1038 0.348 -0.3097 1.201
port 8  -0.3207 0.1124 0.374 -0.2918 1.126
port 9  -0.2981 0.1211 0.401 -0.2739 1.050
port 10 -0.2754 0.1297 0.427 -0.2561 0.975
port 11 -0.2528 0.1383 0.453 -0.2382 0.899
port 12 -0.2302 0.1469 0.480 -0.2203 0.824
port 13 -0.2076 0.1555 0.506 -0.2025 0.749
port 14 -0.1849 0.1641 0.532 -0.1846 0.673
port 15 -0.1623 0.1727 0.559 -0.1667 0.598
port 16 -0.1397 0.1813 0.585 -0.1488 0.522
port 17 -0.1170 0.1899 0.611 -0.1310 0.447
port 18 -0.0944 0.1986 0.637 -0.1131 0.371
port 19 -0.0718 0.2072 0.664 -0.0952 0.296
port 20 -0.0491 0.2158 0.690 -0.0774 0.221
```

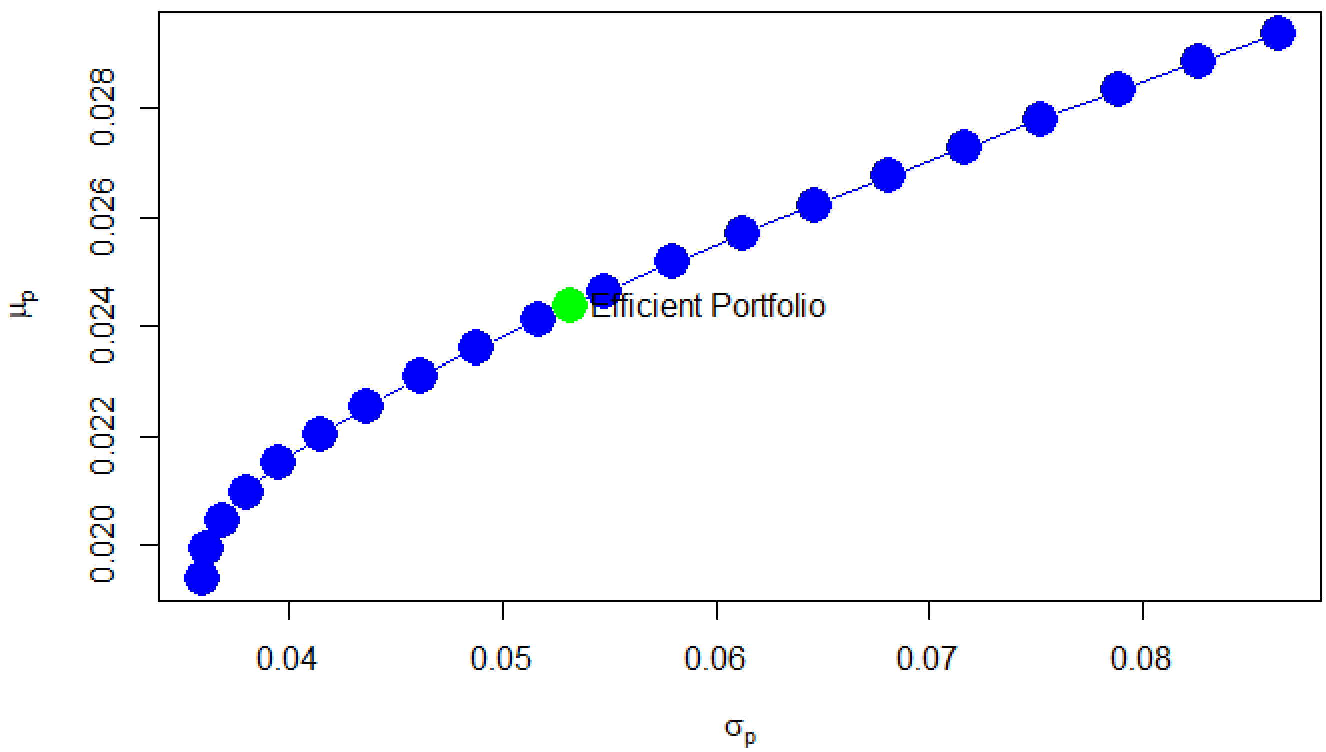
7. Plot the Markowitz bullet based on the efficient portfolios you computed in question 6. On the plot, indicate the location of the minimum variance portfolio and the location of the efficient portfolio found in question 4.

Hide

```
plot(ef$sd, ef$er, type="b", pch=16, col="blue",
     cex= 2.5, ylab=expression(mu[p]), xlab=expression(sigma[p]))
points(sig.px, mu.px, pch=16, col="green", cex=2.5)
```

Hide

```
text(sig.px, mu.px, labels="Efficient Portfolio", pos=4, col = "black")
```



8. Compute the tangency portfolio assuming the risk-free rate is 0.005 ($r_f = 0.5\%$) per month. That is, solve

$$\max_{\mathbf{t}} = \frac{\mathbf{t}'\mu - r_f}{(\mathbf{t}'\Sigma\mathbf{t})^{1/2}}$$

subject to

$$\mathbf{t}'\mathbf{1} = 1$$

where $\mathbf{t}$ denotes the portfolio weights in the tangency portfolio. Are there any negative weights in the tangency portfolio? If so, interpret them.

Hide

```
#computing tangency portfolio weights
one.vec = rep(1, 5)
mu.minus.rf = muhat.vec - r.f * one.vec
top.mat = sigma.inv.mat%*%mu.minus.rf
bot.val = as.numeric(t(one.vec)%*%top.mat)
t.vec = top.mat[,1]/bot.val
t.vec
```

```
  AMZN     BA   COST    JWN   SBUX
-0.112  0.192  0.618 -0.127  0.428
```

Hide

```
#computing ER and sd for tangency portolio
mu.t = as.numeric(crossprod(t.vec, muhat.vec))
sig2.t = as.numeric(t(t.vec)%*%sigma.mat%*%t.vec)
sig.t = sqrt(sig2.t)
mu.t
```

```
[1] 0.0208
```

Hide

```
sig.t
```

```
[1] 0.0377
```

- Again AMZN and JWN are risky assets so it makes sense that there would be short-sales on these assets in the portfolio.

Hide

```
#Double checking
tangency.portfolio(er = muhat.vec, cov.mat = sigma.mat, risk.free = r.f, shorts = T)
```

```
Call:
tangency.portfolio(er = muhat.vec, cov.mat = sigma.mat, risk.free = r.f,
    shorts = T)

Portfolio expected return:     0.0208
Portfolio standard deviation:  0.0377
Portfolio weights:
  AMZN     BA   COST    JWN   SBUX
-0.112  0.192  0.618 -0.127  0.428
```

9. Repeat question 8 but this time do not allow short sales. That is, add the following constraint: $x_i \geq 0 \text{ for } i = AMZN, \cdots SBUX$. Compare the Sharpe ratio of this portfolio with the Sharpe ratio of the tangency portfolio that allows for short sales.

Hide

```
tangency.portfolio(er = muhat.vec, cov.mat = sigma.mat, risk.free = r.f, shorts = F)
```

```
Call:
tangency.portfolio(er = muhat.vec, cov.mat = sigma.mat, risk.free = r.f,
    shorts = F)

Portfolio expected return:     0.0202
Portfolio standard deviation:  0.0379
Portfolio weights:
 AMZN    BA  COST   JWN  SBUX
0.000 0.146 0.496 0.000 0.358
```

Hide

```
sharpe.t.s = (mu.t - r.f) / sig.t
sharpe.t.ns = (0.0202 - r.f) / 0.0379
sharpe.t.s
```

```
[1] 0.421
```

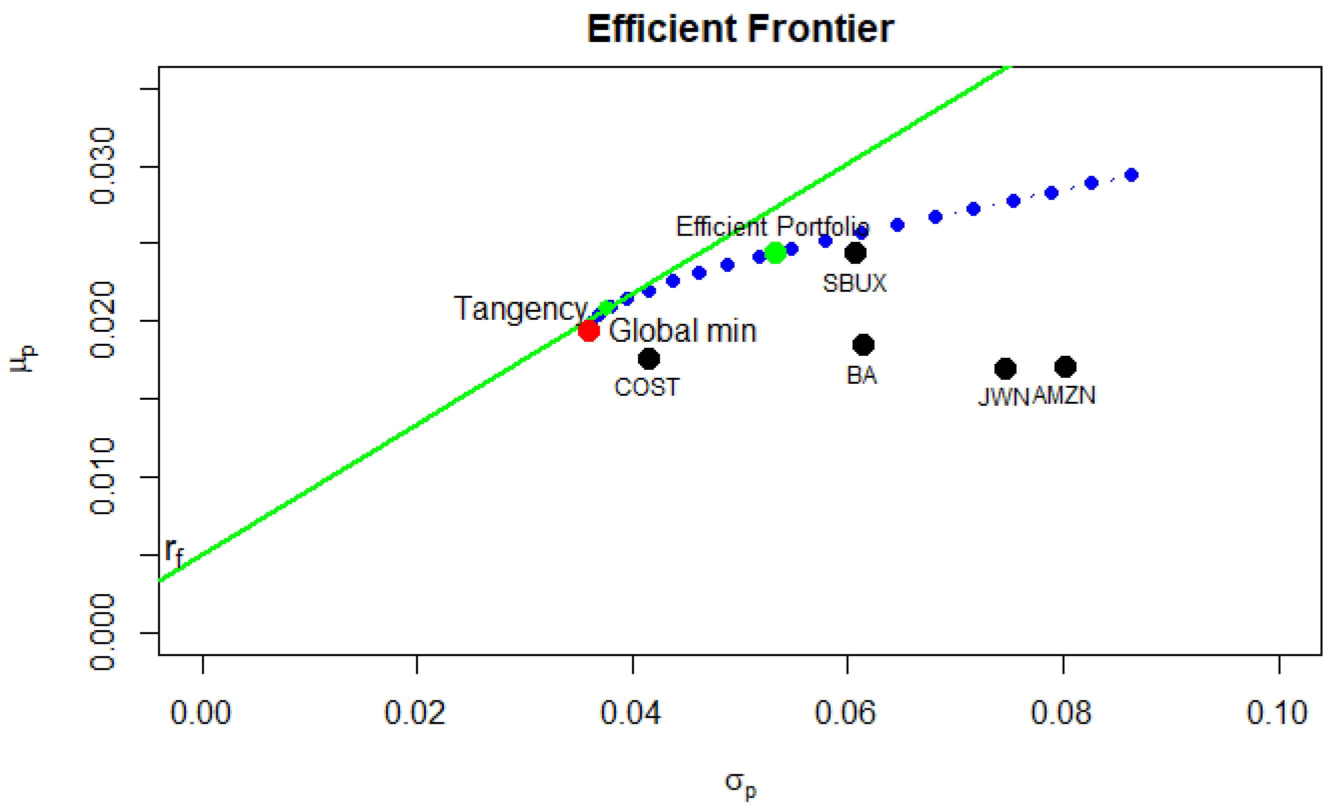
Hide

```
sharpe.t.ns
```

```
[1] 0.401
```

- The sharpe ratio is slightly lower for the tangency portfolio that doesn't allow short-sells.

10. On the graph with the Markowitz bullet, plot the efficient portfolios that are combinations of T-bills and the tangency portfolio that allows short sales. Indicate the location of the tangency portfolio on the graph.



11. Find the efficient portfolio of combinations of T-bills and the tangency portfolio (that allows short sales) that has the same SD value as Starbucks. What is the expected return on this portfolio? Indicate the location of this portfolio on your graph of the Markowitz bullet.

Hide

```
x.t.sbux = sigmahat["SBUX"]/sig.t
x.t.sbux
```

```
SBUX
1.61
```

Hide

```
x.t.sbux * t.vec
```

```
  AMZN     BA   COST    JWN   SBUX
-0.179  0.309  0.994 -0.204  0.690
```

Hide

```
mu.t.sbux = x.t.sbux * mu.t + (1 - x.t.sbux) * r.f
mu.t.sbux
```

```
  SBUX
0.0305
```

Hide

```
sig.t.sbux = x.t.sbux * sig.t
sig.t.sbux
```

```
  SBUX
0.0607
```

Hide

```
plot(ef$sd, ef$er, type="b", pch=16, col="blue",
     cex= 1, ylab=expression(mu[p]), xlab=expression(sigma[p]), ylim = c(0, 0.035), xlim = c(0,
.1), main = "Efficient Frontier")
abline(a=r.f, b=slope.t, col="green", lwd=2)
```

Hide

```
points(sig.t, mu.t, pch=16, col="green", cex=1.15)
points(sd.vec, muhat.vec, pch=16, cex=1.5, col="black")
```

Hide

```
points(sig.t.sbux, mu.t.sbux, pch=16, cex=1.5, col = "purple")
points(sig.gmin, mu.gmin, pch=16, cex=1.5, col="red")
```

Hide

```
points(sig.px, mu.px, pch=16, col="green", cex=1.5)
text(sig.px, mu.px, labels="Efficient Portfolio", pos=3, col = "black", cex = .8)
```

Hide

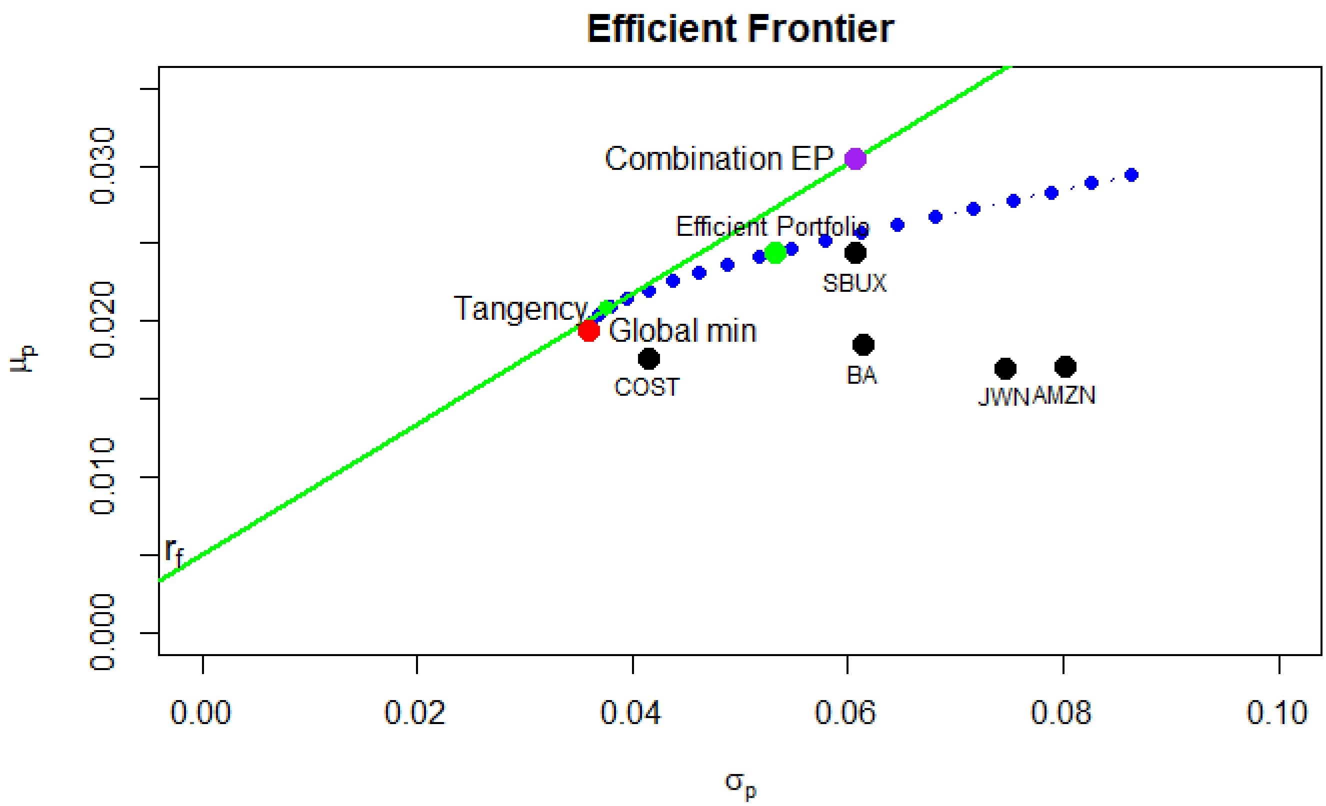
```
text(sig.gmin, mu.gmin, labels="Global min", pos=4, cex=1)
text(sig.t.sbux, mu.t.sbux, labels = "Combination EP", pos=2, cex = 1)
```

Hide

```
text(sd.vec, muhat.vec, labels=asset.names, pos=1, cex=.75)
text(sig.t, mu.t, labels="Tangency", pos=2,cex=1)
```

Hide

```
text(0, rf, labels=expression(r[f]), pos=2, cex=1.15)
```


Efficient Frontier
Combination EP
Efficient Portfolio
Tangency
Global min
SBUX
COST
BA
JWN
AMZN
r_f
μ_p
σ_p
0.000
0.010
0.020
0.030
0.00
0.02
0.04
0.06
0.08
0.10