Project-2-STATC183

Nicholas Davidson

4/17/2023

```
#Read your csv file:
a <- read.csv("stockData.csv", sep=",", header=TRUE)
spy <- read.csv('spy.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)]
#Compute mean vector:
means <- colMeans(r) #Without ^GSPC
#Compute variance covariance matrix:
covmat <- cov(r) #Without ^GSPC
#Compute correlation matrix:
cormat <- cor(r) #Without ^GSPC
#Compute the vector of variances:
variances <- diag(covmat)
#Compute the vector of standard deviations:
stdev <- diag(covmat)^.5
```

(a)

Refer to the lecture material and the paper “An Analytic Derivation of the Efficient Portfolio Frontier,” (JFQA, Robert Merton, 1972).Compute A, B, C, D

```
ones <- rep(1, ncol(r))
A <- t(ones) %*% solve(covmat) %*% means
B <- t(means) %*% solve(covmat) %*% means
C <- t(ones) %*% solve(covmat) %*% ones
D <- B*C-A^2
```

(b)

Compute the values of $\lambda_1 \mathrm{andlambda}_2$ (the two Lagrange multipliers).

```
# Suppose E = 0.02
E1=0.02

lambda1 <- (C*E1-A)/D
lambda2 <- (B-A*E1)/D
```

(c)

Suppose an investor has a prescribed expected return E. Find the composition of the efficient portfolio given the return E.

```
X <- solve(covmat) %*%(lambda1*means + lambda2*ones)
```

```
## Warning in lambda1 * means: Recycling array of length 1 in array-vector arithmetic is deprecated.
##   Use c() or as.vector() instead.
```

```
## Warning in lambda2 * ones: Recycling array of length 1 in array-vector arithmetic is deprecated.
##   Use c() or as.vector() instead.
```

```
X
```

```
##                [,1]
## AAPL  -0.097703520
## MSFT  -0.037860534
## NVDA   0.042912141
## CSCO   0.075938588
## INTC   0.112865552
## MU    -0.029834715
## BRK.B -0.054042715
## V      0.114126750
## JPM    0.359675943
## GS    -0.108240544
## BLK   -0.178535544
## BX    -0.039765453
## COST  -0.057389866
## KO     0.226443343
## SYY    0.354776262
## CHGG   0.042469184
## PG     0.367006620
## TGT    0.069295252
## AMZN  -0.041869685
## SBUX   0.150963251
## F     -0.140219210
## GM    -0.004920857
## LULU  -0.027828756
## NKE    0.001974496
## GOOGL -0.088564672
## META   0.173736599
## VZ    -0.113134851
## T      0.050139615
## CMCSA -0.215648745
## DIS    0.093236073
```

(d)

Use your data to plot the frontier in the mean-variance space (parabola)

```
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.
```

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

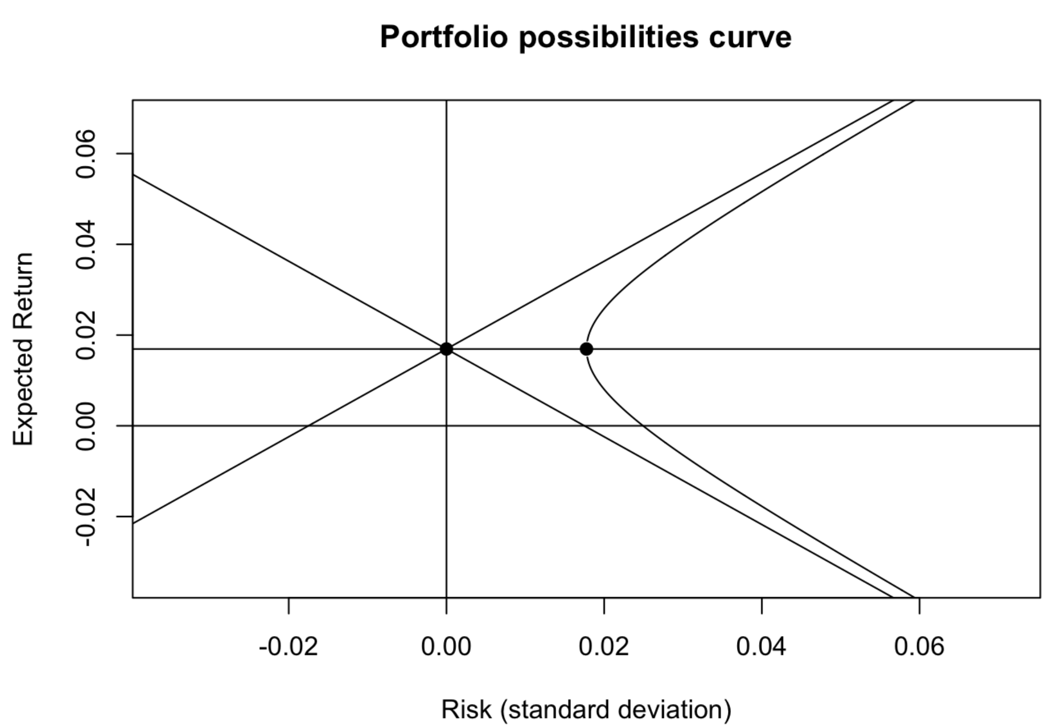
```
## Warning in V * sqrt(D/C): Recycling array of length 1 in vector-array arithmetic is deprecated.
##   Use c() or as.vector() instead.
```

```
## 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.
```

```
    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.
```

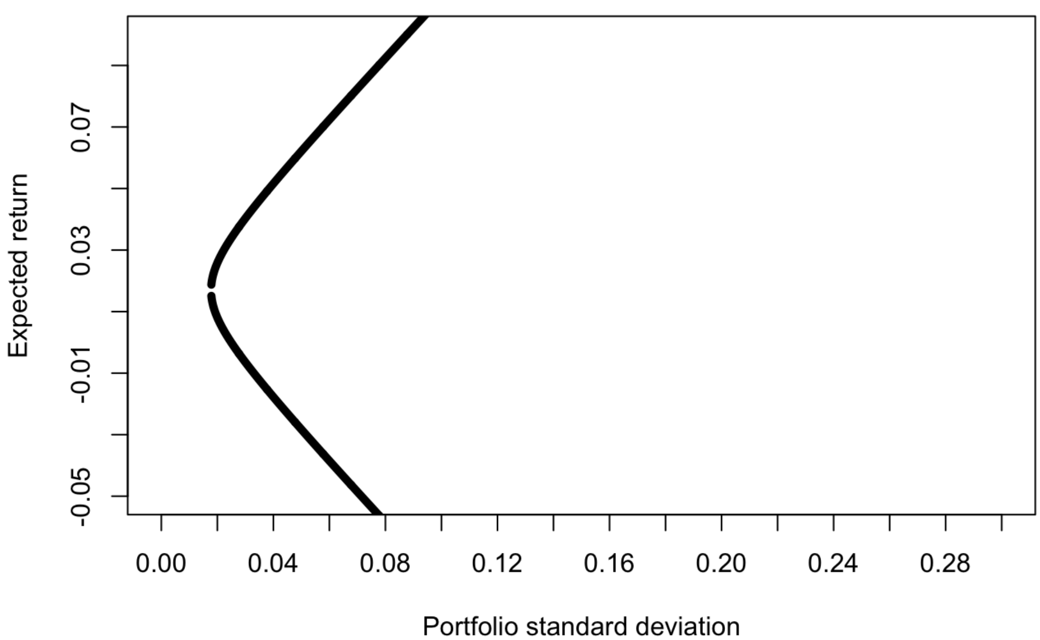
```
    options(warn = -1)
    y1 <- (A + sqrt(D*(C*sdeff^2 - 1)))*(1/C)
    y2 <- (A - sqrt(D*(C*sdeff^2 - 1)))*(1/C)
   # options(warn = 0)

    points(sdeff, y1, type = "l")
    points(sdeff, y2, type = "l")
```



(e)

Use your data to plot the frontier in the mean-standard deviation space using the hyperbola method

```
#Hyperbola:
#Efficient frontier:
    minvar <- 1/C
    minE <- A/C
    sdeff <- seq((minvar)^0.5, 1, by = 0.0001)
#    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)

plot(sdeff, y1, type = "n",xlim=c(0,0.3), ylim=c(-0.05,0.10), xlab="Portfolio standard deviation", ylab="Expe
cted return", xaxt="no", yaxt="no")

axis(1, at=seq(0, 0.3, 0.02))
axis(2, at=seq(-0.05,0.10, 0.02))

    points(sdeff, y1, lwd=5,type = "l")
    points(sdeff, y2, lwd=5,type = "l")
```

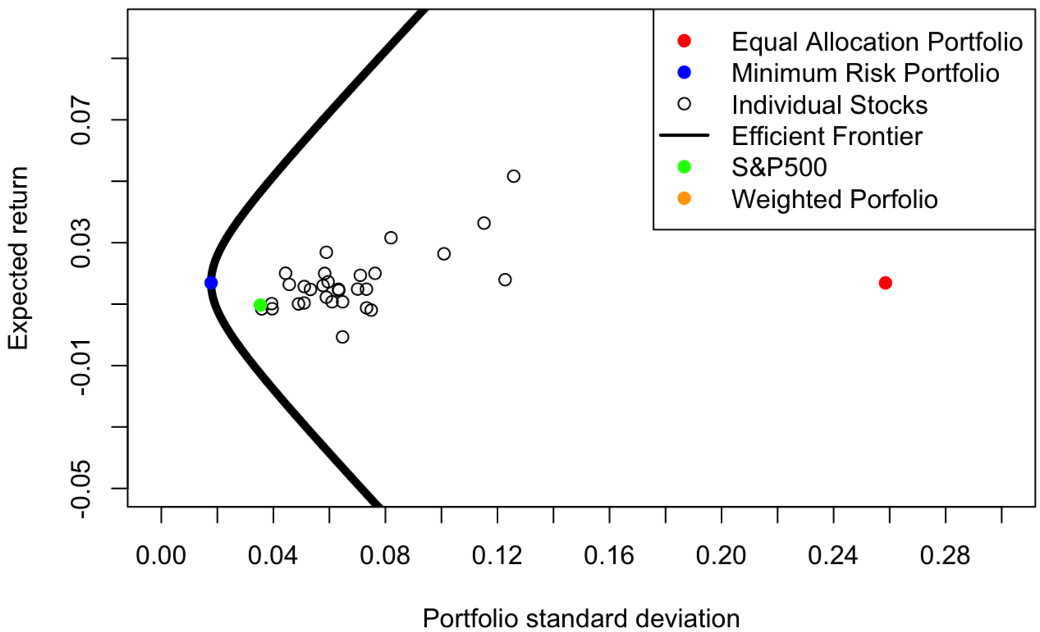


(f)

On the plot in (e) add the 30 stocks, the S&P500, the equal allocation portfolio, the minimum risk portfolio, and the portfolio in (c).

```
# Plot in (e)
#Hyperbola:
#Efficient frontier:
    minvar <- 1/C
    minE <- A/C
    sdeff <- seq((minvar)^0.5, 1, by = 0.0001)
#    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)

plot(sdeff, y1, type = "n",xlim=c(0,0.3), ylim=c(-0.05,0.10), xlab="Portfolio standard deviation", ylab="Expe
cted return", xaxt="no", yaxt="no")

axis(1, at=seq(0, 0.3, 0.02))
axis(2, at=seq(-0.05,0.10, 0.02))

    points(sdeff, y1, lwd=5,type = "l")
    points(sdeff, y2, lwd=5,type = "l")

# Plot the equal allocation portfolio
means <- colMeans(r)
covmat <- cov(r)
variances <- diag(covmat)^.5
mean.port <- sum(means * 1/ncol(r))
sd.port <- sqrt(sum(variances * 1/ncol(r)))
points(sd.port, mean.port, col="red", pch=19)

# Plot the minimum Risk Portfolio
points(sqrt(1/C), (A/C), col="blue", pch=19)

# Plot the 30 Stocks
#Convert adjusted close prices into returns:
r <- (a[-1,3:ncol(a)]-a[-nrow(a),3:ncol(a)])/a[-nrow(a),3:ncol(a)]
#Compute mean vector:
means <- colMeans(r) #Without ^GSPC
#Compute variance covariance matrix:
covmat <- cov(r) #Without ^GSPC
#Compute correlation matrix:
cormat <- cor(r) #Without ^GSPC
#Compute the vector of variances:
variances <- diag(covmat)
#Compute the vector of standard deviations:
stdev <- diag(covmat)^.5
points(stdev, means)

# Plot the S&P 500
returns <- diff(log(spy$SPY))
#Compute mean vector:
means <- mean(returns) #Without ^GSPC
#Compute variance:
variance <- var(returns)

#Compute the vector of standard deviations:
stdev <- sqrt(variance)
points(stdev, means, col = 'green', pch = 19)

### Returns for a given return value
#Compute mean vector:
means <- t(X) %*% colMeans(r) #Without ^GSPC
#Compute variance covariance matrix:
covmat = cor(r)
risk <- sqrt(diag(t(X) %*% covmat %*% X))
```

```
points(risk, means, col = "orange", pch = 19)


# Add a legend
legend("topright", legend=c("Equal Allocation Portfolio", "Minimum Risk Portfolio", "Individual Stocks", "Eff
icient Frontier", "S&P500", 'Weighted Porfolio'), col=c("red", "blue", "black", "black", "green", "orange"),
pch=c(19, 19, 1, NA, 19, 19), lty=c(NA, NA, NA, 1, NA, NA), lwd=c(NA, NA, NA, 2, NA, NA))
```

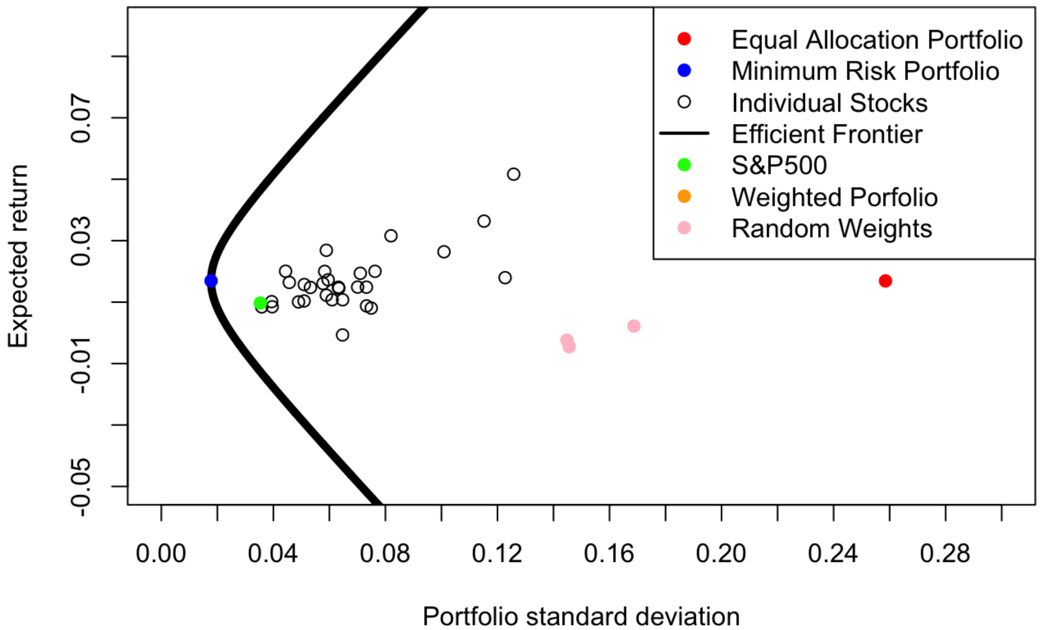


(g)

Add three arbitrary portfolios on the plot of (c). You can choose any 30 weights with $\sum_{i=1}^{30} x_i = 1$

```
# repeat part (f)
# Plot in (e)
#Hyperbola:
#Efficient frontier:
    minvar <- 1/C
    minE <- A/C
    sdeff <- seq((minvar)^0.5, 1, by = 0.0001)
#    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)

plot(sdeff, y1, type = "n",xlim=c(0,0.3), ylim=c(-0.05,0.10), xlab="Portfolio standard deviation", ylab="Expe
cted return", xaxt="no", yaxt="no")

axis(1, at=seq(0, 0.3, 0.02))
axis(2, at=seq(-0.05,0.10, 0.02))

    points(sdeff, y1, lwd=5,type = "l")
    points(sdeff, y2, lwd=5,type = "l")

# Plot the equal allocation portfolio
means <- colMeans(r)
covmat <- cov(r)
variances <- diag(covmat)^.5
mean.port <- sum(means * 1/ncol(r))
sd.port <- sqrt(sum(variances * 1/ncol(r)))
points(sd.port, mean.port, col="red", pch=19)

# Plot the minimum Risk Portfolio
points(sqrt(1/C), (A/C), col="blue", pch=19)

# Plot the 30 Stocks
#Convert adjusted close prices into returns:
r <- (a[-1,3:ncol(a)]-a[-nrow(a),3:ncol(a)])/a[-nrow(a),3:ncol(a)]
#Compute mean vector:
means <- colMeans(r) #Without ^GSPC
#Compute variance covariance matrix:
covmat <- cov(r) #Without ^GSPC
#Compute correlation matrix:
cormat <- cor(r) #Without ^GSPC
#Compute the vector of variances:
variances <- diag(covmat)
#Compute the vector of standard deviations:
stdev <- diag(covmat)^.5
points(stdev, means)

# Plot the S&P 500
returns <- diff(log(spy$SPY))
#Compute mean vector:
means <- mean(returns) #Without ^GSPC
#Compute variance:
variance <- var(returns)

#Compute the vector of standard deviations:
stdev <- sqrt(variance)
points(stdev, means, col = 'green', pch = 19)

### Returns for a given return value
#Compute mean vector:
means <- t(X) %*% colMeans(r) #Without ^GSPC
#Compute variance covariance matrix:
covmat = cor(r)
```

```
risk <- sqrt(diag(t(X) %*% covmat %*% X))
points(risk, means, col = "orange", pch = 19)


# random weights
w <- rnorm(30)
# normalize weights
w <- w/30
# compute portfolio returns
means <- colMeans(r)
r.port <- t(w) %*% means
# compute portfolio risk
sd.port <- sqrt(diag(t(w) %*% covmat %*% w))
# plot the portfolio returns vs risk
points(sd.port, r.port, col = "pink", pch = 19)

# random weights
w <- rnorm(30)
# normalize weights
w <- w/30
# compute portfolio returns
r.port <- t(w) %*% means
# compute portfolio risk
sd.port <- sqrt(diag(t(w) %*% covmat %*% w))
# plot the portfolio returns vs risk
points(sd.port, r.port, col = "pink", pch = 19)

# random weights
w <- rnorm(30)
# normalize weights
w <- w/30
# compute portfolio returns
r.port <- t(w) %*% means
# compute portfolio risk
sd.port <- sqrt(diag(t(w) %*% covmat %*% w))
# plot the portfolio returns vs risk
points(sd.port, r.port, col = "pink", pch = 19)

legend("topright", legend=c("Equal Allocation Portfolio", "Minimum Risk Portfolio", "Individual Stocks", "Eff
icient Frontier", "S&P500", 'Weighted Porfolio', "Random Weights"), col=c("red", "blue", "black", "black", "g
reen", "orange", "pink"), pch=c(19, 19, 1, NA, 19, 19), lty=c(NA, NA, NA, 1, NA, NA), lwd=c(NA, NA, NA, 2, N
A, NA))
```


Equal Allocation Portfolio
Minimum Risk Portfolio
Individual Stocks
Efficient Frontier
S&P500
Weighted Porfolio
Random Weights
Expected return
-0.05
-0.01
0.03
0.07
0.00
0.04
0.08
0.12
0.16
0.20
0.24
0.28
Portfolio standard deviation