Assignment 2 – Mean-Variance Optimization Hands-on Exercise

Contents

Final Deliverable

You may use any programming language that you prefer. In your final submission, answer the following questions:

1. How many rows of data do you delete in the price matrix? How many rows does the return matrix have? (5pt).
2. Is the covariance matrix for the 50 selected stocks positive definite? (5pt)
3. What are the mean return and variance for the minimum risk portfolio constructed from the 50 selected risky assets? (10pt)
4. Suppose you have $1,000 and would like to invest in the minimum risk portfolio, how much do you invest in IBM? (5pt)
5. If your risk-tolerance parameter is 0.1, what are the mean return and variance of the parametric efficient portfolio that's suitable for you? (10pt)
6. Suppose you have $1,000 and would like to invest in the parametric-efficient portfolio, how much do you invest in IBM? (5pt)
7. Plot an efficient frontier plot in the (σ, μ)-plane along with the 50 individual assets. (10pt)

Executive summary

In this exercise, we will see the following practical problems in financial analysis:

- Missing data: Two of the stocks in the dataset has some missing data. This is troublesome in many ways (for example, cannot estimate covariance matrix).
  - A simplistic solution is to truncate the dataset to a narrower observation window where all stocks have valid prices.

- Estimated covariance matrix is NOT positive definite, which may seem strange at first. There are many ways this can happen. In our case, this is because we have more stocks (about 500) than available data (about 200).
    - In practice, you may want to find ways to collect more data.
    - In this assignment, we consider mean-variance portfolio optimization for a small subset of the stocks.

Load data

The following codes show how you can clear temporary variables (e.g., some variables left from previous codes) and load data from the .csv file provided. The first ten rows and ten columns of the loaded data are displayed.

```r
rm(list=ls())      # clear temporary variables
library(ggplot2)  # library for plotting

prices <- read.csv("HistoricalPrices.csv",header = T)    # read data from file
prices[1:10,1:10] # visualize the historical price data
```

```
##     X   ref.date        A      AAL      AAP     AAPL     ABBV      ABC   ABMD
## 1   1 2020-01-02 85.25838 28.98289 157.9097 74.44460 83.87167 83.59086 168.81
## 2   2 2020-01-03 83.88947 27.54819 157.9196 73.72084 83.07555 82.53960 166.82
## 3   3 2020-01-06 84.13747 27.21941 155.3243 74.30827 83.73117 83.74805 179.04
## 4   4 2020-01-07 84.39538 27.11978 153.4818 73.95879 83.25351 83.14873 180.35
## 5   5 2020-01-08 85.22862 27.73749 151.7186 75.14852 83.84357 83.95437 178.69
## 6   6 2020-01-09 86.56775 27.84709 151.4313 76.74473 84.48981 85.14320 183.60
## 7   7 2020-01-10 86.88517 27.21941 147.6076 76.91822 83.41273 85.51655 189.06
## 8   8 2020-01-13 86.75623 27.28915 143.8929 78.56153 82.90698 84.64212 168.10
## 9   9 2020-01-14 87.28196 27.42864 147.4987 77.50070 83.72332 87.27522 172.73
## 10 10 2020-01-15 87.90689 27.47845 148.7964 77.16856 84.72953 89.77076 177.92
##         ABT
## 1  85.25698
## 2  84.21763
## 3  84.65886
## 4  84.18821
## 5  84.53140
## 6  84.75692
## 7  83.69795
## 8  83.46262
## 9  84.42760
## 10 86.04251
```

Preliminary data manipulation (missing data particularly)

Seeing the first ten columns and rows of the "price" matrix above, we determine that the first two columns are irrelevant so we can remove them:

Suppose A is a matrix in R, then $A[i, j]$ calls the (i, j)-th element of the matrix. Also, $A[i,]$ and $A[, j]$ calls the i-th row and the j-th column of the matrix. The minus ('-') sign in "prices[,-c(1,2)]" tells R to remove Column 1 & Column 2 of the "prices" matrix.

Your tasks:

1. Take a look at the columns of the "prices" matrix with titles "CARR" and "OTIS", you should see some NAs, i.e., missing data.

- This is because CARR and OTIS joined S&P 500 in March 2020. So these two stocks' price data is available only March 20.

2. Remove just enough rows of the "prices" matrix so that all the NAs in these two stocks are removed.

- By removing the rows, you will be removing some stock prices for other stocks too.
- Such removal makes sure that all the stocks have the same amount of data for later analysis. This is a simplistic solution to dealing with missing data.
- When you are done, the first stock price for "CARR" is $11.89012.

3. Calculate the returns matrix based on the resulting prices matrix.

- When you are done, the first three returns for "Stock A" are, 0.01064475, -0.04649420, and 0.05728266.
- The returns matrix should have 1 less row than

```r
prices <- prices[,-c(1,2)]    # remove the first two irrelevant columns

# complete the following line
# Your codes may be longer than 1 line, but please use the variable name "prices".
prices <- prices[-c(1:53),]   # remove some rows so all stock prices are available

# complete the following line
# Your codes may be longer than 1 line, but please use the variable name "returns".
returns <- as.data.frame(diff(as.matrix(prices)))/prices[-nrow(prices),]
```

Stock returns and risks

There is nothing you need to complete in this section. This section is simply a re-run of our synchronous time activity on Jan 18, 2021. This time, the missing data is removed.

Indeed we see a positive slop, i.e., risk-reward trade-off. But again, what we see highly depends on the observation window and the observation frequency of the stock data that we use.

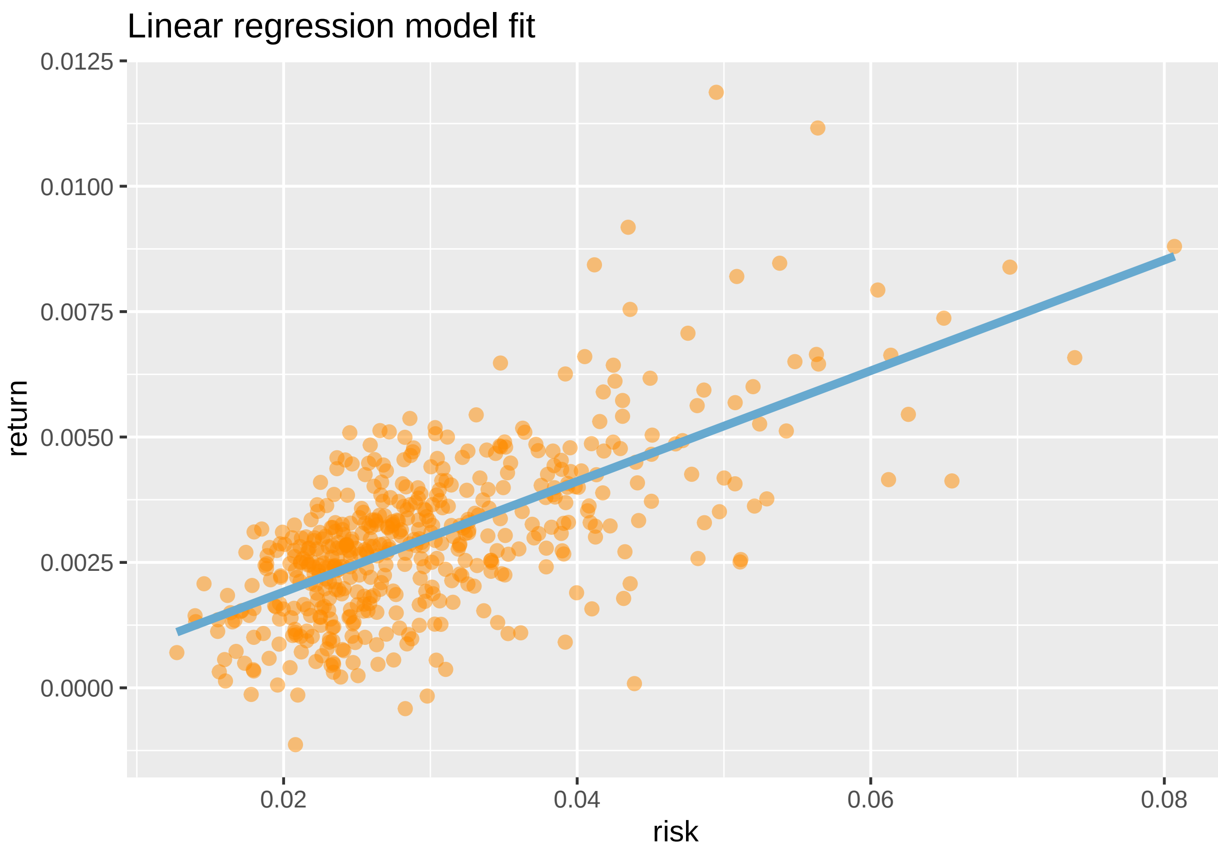
```r
# mean return and standard deviation
mean.return <- colMeans(returns)
std.return <- apply(returns, 2, sd)
summary(lm(mean.return ~ std.return))
```

```
##
## Call:
## lm(formula = mean.return ~ std.return)
##
## Residuals:
##        Min         1Q     Median         3Q        Max
## -0.0044625 -0.0008479  0.0000652  0.0007331  0.0067115
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.0002943  0.0001773   -1.66   0.0976 .
## std.return   0.1102628  0.0056416   19.55   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.001251 on 499 degrees of freedom
## Multiple R-squared:  0.4336, Adjusted R-squared:  0.4325
## F-statistic:   382 on 1 and 499 DF,  p-value: < 2.2e-16
```

```r
# Organize data for plotting
Data <- data.frame(return = mean.return, risk = std.return)

# Plot scatter plot of stock returns vs. risks & line of best fit
p1 <- ggplot(Data, aes(y=return, x=risk)) +
  geom_point(alpha = 0.5, size=2, col= "darkorange") +
  geom_smooth(formula = y~x,method = "lm", se=F, lwd=1.5, col="#67a9cf") +
  ggtitle("Linear regression model fit")

p1
```



Mean-variance Optimization – First Attempt

There is nothing you need to complete in this section.

Please pay attention to how the mean return vector and the covariance matrix are estimated. A new library is loaded to check if the estimated covariance matrix is positive definite, which is an assumption required by mean-variance optimization.

The answer turns out to be NO?!

- One reason for non-positive-definite covariance matrix is that we have more stocks (about 500) than data (about 200).
- One consequence of non-positive-definite covariance matrix is that it is not invertible.
- We need the inverse of the covariance matrix to calculate the optimal portfolios, as we will see later.

```r
mu.vec <- matrix(mean.return, ncol = 1) # mean vector, make sure it is a column vector
Sigma.mat <- cov(returns)                # covariance matrix

library(matrixcalc)   # a library to check if a matrix is positive definite
is.positive.definite(Sigma.mat) # the covariance matrix fails the positive definite test
```

```
## [1] FALSE
```

```r
Sig.inv <- solve(Sigma.mat) # invert the covariance matrix, or report error if not possible
```

```
## Error in solve.default(Sigma.mat): system is computationally singular: reciprocal condition number =
```

Mean-variance Optimization: Preliminary Analysis

We will next consider a mean-variance optimization problem for 50 stocks instead. With less stocks to consider, the estimated convariance matrix is now positive definite. As a result, we can calculate the optimal portfolios and visualize the efficient frontier.

Your tasks:

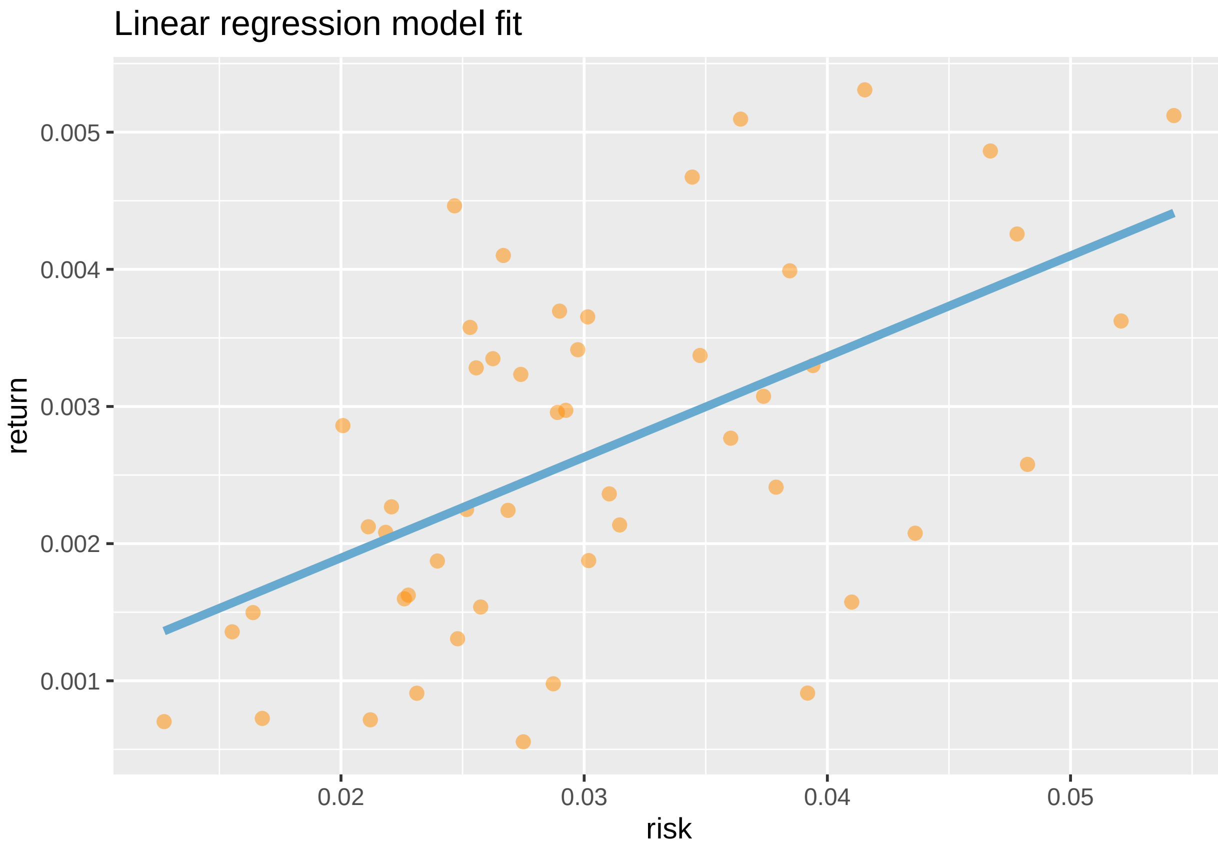
1. Calculate the returns of the 50 stocks specified in StockNames.csv.
2. Estimate the returns and standard deviations of the 50 stocks specified in StockNames.csv.
3. Plot the risk vs. return for the 50 stocks and the resulting line-of-best-fit. *This step is done for you if you keep the variable names as suggested.*
4. Calculate the mean vector and covariance matrix for the 50 stocks. Then verify if the covariance matrix is positive definite.

```r
stocks <- read.csv("StockNames.csv")[,2]  # the second column contains stock names

# complete the following lines
returns <- returns[,stocks]
mean.return <- colMeans(returns)
std.return <- apply(returns, 2, sd)

Data <- data.frame(return = mean.return, risk = std.return)

ggplot(Data, aes(y=return, x=risk)) +
    geom_point(alpha = 0.5, size=2, col= "darkorange") +
    geom_smooth(formula = y~x,method = "lm", se=F, lwd=1.5, col="#67a9cf") +
    ggtitle("Linear regression model fit")
```




```
# complete the following line
mu.vec <- matrix(mean.return, ncol = 1) # mean vector, make sure it is a column vector
Sigma.mat <- cov(returns)                # covariance matrix

is.positive.definite(Sigma.mat) # the covariance matrix fails the positive definite test
```

```
## [1] TRUE
```

As we can see, the covariance matrix passes the positive definiteness test.

Mean-Variance Optimization: Optimal Portfolios

In this Section, you will be asked to use the formulas we derived in class to performn mean-variance portfolio optimization and answer a few questions based on your calculations.

Your tasks:

1. Calculate the minimum risk portfolio as well as its mean return and standard deviation.
2. Based on your calculations, answer Questions 3 & 4 in the "Final Deliverable" section.
3. Calculate the "zero-covariance" portfolio as well as its mean return and standard deviation.
4. Based on your calculations, answer Questions 5 & 6 in the "Final Deliverable" section.

```
Sigma.inv <- solve(Sigma.mat)      # calculate the inverse of the covariance matrix
ell.vec <- matrix(1,ncol = 1, nrow = 50) # define a column vector of 1's

Sigma.ell <- Sigma.inv %*% ell.vec                      # maxtrix-vector product
ell.Sigma.ell <- as.numeric(t(ell.vec) %*% Sigma.ell) # inner product of vectors
# The "as.numeric" may seems strange. But in R, dividing an 1-by-1 matrix is an
```

```r
# invalid calculation. So we need to translate the 1-by-1 matrix into a number first.

##################################
# complete the following lines #
##################################
x.m <- Sigma.ell / ell.Sigma.ell            # min-risk portfolio
mu.m <- as.numeric(t(x.m) %*% mu.vec)       # mean return of min-risk portfolio
sigma2.m <- as.numeric(t(x.m) %*% Sigma.mat %*% x.m)  # variance of min-risk portfolio
mu.m
```

```
## [1] 0.0007479783
```

```r
sigma2.m
```

```
## [1] 7.232177e-05
```

```r
x.m["IBM",]
```

```
##         IBM
## -0.06548979
```

```r
Sigma.mu <- Sigma.inv %*% mu.vec
ell.Sigma.mu <- as.numeric(t(ell.vec) %*% Sigma.mu)

x.z <- Sigma.mu - (ell.Sigma.mu) * x.m
mu.z <- as.numeric(t(x.z) %*% mu.vec)
sigma2.z <- as.numeric(t(x.z) %*% Sigma.mat %*% x.z)

tau <- 0.1
x.tau <- x.m + tau*x.z
mu.tau <- as.numeric(t(x.tau) %*% mu.vec)
sigma2.tau <- as.numeric(t(x.tau) %*% Sigma.mat %*% x.tau)
mu.tau
```

```
## [1] 0.01402798
```

```r
sigma2.tau
```

```
## [1] 0.001400322
```

```r
x.tau["IBM",]
```

```
##        IBM
## -0.8614289
```

Mean-Variance Optimization: Efficient Frontier

In this Section, you will be asked to use the optimal portfolios calculated above to plot the efficient frontier.

Your tasks:

1. Based on the set of risk-tolerance parameters provided, calculate the corresponding parametric efficient portfolios' mean returns and variances.
   - Hint: You can, but do not have to calculate the optimal portfolios first.
2. Plot the efficient frontier in the (σ, μ)-plane along with the 50 individual assets. Also indicate the parametric-efficient portfolio for $\tau = 0.1$ in the plot.

```r
tau <- seq(0, 0.15, length.out = 100)
```

```
###############################
# complete the following lines #
###############################
mu.p <- mu.m + tau * mu.z
sigma2.p <- sigma2.m + tau^2 * sigma2.z

Data2 <- data.frame(mu = mu.p, sigma = sqrt(sigma2.p))
Data3 <- data.frame(mu = mu.tau, sigma = sqrt(sigma2.tau))

ggplot(Data, aes(y=return, x=risk)) +
    geom_point(alpha = 0.5, size=2, col= "darkorange") +
    geom_line(data = Data2, aes(x=sigma, y=mu), color = "#67a9cf", lwd = 1.5) +
    geom_point(data = Data3, aes(x=sigma, y=mu), color = "red", size = 5, shape=13) +
    ggtitle("Individual assets and risky efficient frontier")
```