Apply Markovitz model to optimize portfolio investment in stock trading

Than Doan Thuan[1], Ngo Phuong Nam[1], Bui Doan Khanh Quan[1], Nguyen Dang Tien Dung[1], and Nguyen Hoang Ngoc Ha[1]

[1]Undergraduate student

[2]Fulbright University Vietnam, Ho Chi Minh city, Vietnam.

[†]These authors contributed equally to this work.

Abstract

This paper is an attempt to devise an optimal approach to calculating the suitable and precise amount of assets to be allocated to a number of given securities. Based on the expected return of each security, the price data set will be converted into the covariance matrix with some incorporation of derivatives with a view to maximizing the income gained from stock exchanging. The input is the historical price of a number of n securities pre-chosen by the investors. By applying the principles of linear algebra as well as some properties from calculus and statistics, the main goal of this research is to compute the optimal weight of each security of the investors. To find the optimized portfolio, the Lagrange multipliers function and the Mean-Variance models of Markowitz will be used to find the weight vectors. The optimal result indicates the weight with the highest ratio.

1 Introduction

Stock investment has been rising in popularity recently due to the advancement of online technology, as well as because of the pandemic which has resulted in people spending more time in their homes, might also be another reason for their tilt towards the stock market trading. A relevant reference for stock market investors about the selection of the appropriate stock and a recommendation for the amount of assets to be invested in the right stock with the right amount of assets to maximize their income is desired by many amateur stock investors.

Previously, the strategy of equal-weighted distribution, in which investors would distribute an equal amount of assets to every security chosen security, was thought to be the most fruitful approach. However, due to the differences in the fluctuations of the market price of different stocks, equal-weighted distribution could not ensure the optimal return for the investors.

Instead of gaining maximal profit by the method of equal-weighted distribution or fundamental analysis, investors can adopt the possibly more beneficial method of portfolio analysis, or portfolio optimization. In this method, the Mean-Variance models of Markowitz of calculating the optimal weight vector will be used. From then we can form matrix transformations to find the maximum weight needed to invest into the securities.

2 Methods

2.1 Terminologies

2.1.1 Stock returns:

It is defined by the equation:

$$r_t = \frac{P_t}{P_{t-1}}$$

While P_t means "price at time t", R_t is a shorthand for return between time t-1 and t. R indicates whether the price is increasing or decreasing. If $r > 1$, prices increases. If $r < 1$, prices decreases.
In this problem, suppose P_{it} stock price t at time t, and r_{it} stock return i at time t. The value of r_{it} can be calculated using the following equation:

$$r_{it} = \frac{P_{it}}{P_{it-1}}$$

where i=1,...,N with N number of stocks that were analyzed, and t=1,...,T with T the number of stock price data observed.

2.1.2 Expected return:

The expected return is the amount of profit or loss an investor can anticipate receiving on an investment. It is calculated by the historic average of a stock's return over a given period. To calculate for a portfolio of securities, we will use calculation related to the weighted average of the expected individual returns:

$$\text{Expected return} = \sum x_i w_i$$

$$x_i = \text{expected return at } i^{th}\text{asset}$$

$$w_i = \text{weight or portion of } i^{th}\text{asset in the portfolio}$$

2.1.3 Covariance

Covariance measures how much variables change together. It is a statistical measure to address the interrelationship between the returns of two securities. To calculate covariance:

$$cov\,(x, y) = \frac{\sum (x_i - \overline{x})\,(y_i - \overline{y})}{N}$$

$$x_i = \text{data value of x}, y_i = \text{data value of y}$$

$$\overline{x} = \text{mean of x}, \overline{y} = \text{mean of y}$$

$$N = \text{number of data values}$$

2.1.4 Covariance matrix

Covariance matrix is a square matrix that shows the covariance between each pair of entries of a given random vector in the matrix. Thus, this matrix calculates the distribution of data in the data sets and show how large the changes in data of the dataset.

So we suppose the covariance matrix based on r_{it}, r_{jt}:

$$\sum = \begin{pmatrix} \sigma_1^2 & \sigma_{12} & \dots & \sigma_{1N} \\ \sigma_{21} & \sigma_2^2 & \dots & \sigma_{2N} \\ M & M & M & M \\ \sigma_{N1} & \sigma_{22} & \dots & \sigma_N^N \end{pmatrix} \text{ where } \sigma_i = Cov(r_{it}, r_{jt})$$

2.1.5 Lagrange multiplier function

In mathematical optimization, the method of Lagrange multipliers is a strategy for finding the local maxima and minima of a function subject to equality constraints. This technique introduces a third variable (the multiplier) that enables you to solve constrained optimization problems without first solving the constraint equation for one of the variables.

The method of Lagrange multipliers[**2**] :

Suppose that f(x, y, z) and g(x, y, z) are differentiable and ∇ g $\neq 0$ when g(x, y, z) = 0. To find the local maximum and minimum values of f subject to the constraint g(x, y, z) = 0 (if these exist), find the values of x, y, z, and λ that simultaneously satisfy the equations:

$$\nabla f = \lambda \nabla \text{g and } g(x, y, z) = 0$$

2.2 Mathematical model

Suppose r_i stock return i with N number $1, 2, 3, ..., N$ of stocks were analyzed. We denote some vectors as below :

a) The weight vector $w' = \begin{pmatrix} w_1, w_2, w_3, ..., w_n \end{pmatrix}$

b) The vector of stock return $r' = \begin{pmatrix} r_1 \\ r_2 \\ r_3 \\ ... \\ r_n \end{pmatrix}$

c) The unit vector $e' = \begin{pmatrix} 1, 1, 1, 1, 1, 1, ..., 1 \end{pmatrix}$,

Then, the portfolio return can be expressed as:

$$r_p = \begin{pmatrix} w_1, w_2, w_3, ..., w_n \end{pmatrix} \begin{pmatrix} r_1 \\ r_2 \\ r_3 \\ ... \\ r_n \end{pmatrix} \text{ where } \begin{pmatrix} w_1, w_2, w_3, ..., w_n \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 1 \\ ...1 \end{pmatrix} = 1$$

Based on (1), suppose $\mu' = (\mu_1, \mu_2, ..., \mu_N)$, expectation of portfolio μ_p can be expressed as :

$$u_p = E[r_p] = (w_1, w_2, w_3, ..., w_n) \begin{pmatrix} \mu_1 \\ \mu_2 \\ ... \\ \mu_N \end{pmatrix} \tag{1}$$

From 3.1.4, we get the covariance matrix of the returns.

$$\sum = \begin{pmatrix} \sigma_1^2 & \sigma_{12} & \dots & \sigma_{1N} \\ \sigma_{21} & \sigma_2^2 & \dots & \sigma_{2N} \\ M & M & M & M \\ \sigma_{N1} & \sigma_{22} & \dots & \sigma_N^N \end{pmatrix} \text{ where } \sigma_i = Cov(r_{it}, r_{jt}) \tag{2}$$

Then, variance of the portfolio return can be expressed as follows:

$$\sigma_p^2 = \begin{pmatrix} w_1 w_2 w_3 .... w_N \end{pmatrix} \begin{pmatrix} \sigma_1^2 & \sigma_{12} & \dots & \sigma_{1N} \\ \sigma_{21} & \sigma_2^2 & \dots & \sigma_{2N} \\ ... & ... & ... & ... \\ \sigma_{N1} & \sigma_{22} & \dots & \sigma_N^N \end{pmatrix} \begin{pmatrix} w_1 \\ w_2 \\ w_3 \\ ... \\ w_N \end{pmatrix}$$

The common interpretation is we should maximize return given a level of risk or we should minimize risk given a level of return [**1**]. So, with given level of risk τ, this problem, which is known as the maximum value of expected portfolio utility, means we have to find the maximum value of :

$$2\tau(w_1, w_2, w_3, ..., w_n)\begin{pmatrix}\mu_1\\ \mu_2\\ ...\\ \mu_N\end{pmatrix} - (w_1, w_2, ..., w_n)\begin{pmatrix}\sigma_1^2 & \sigma_{12} & \dots & \sigma_{1N}\\ \sigma_{21} & \sigma_2^2 & \dots & \sigma_{2N}\\ ... & ... & ... & ...\\ \sigma_{N1} & \sigma_{22} & \dots & \sigma_N^N\end{pmatrix}\begin{pmatrix}w_1\\ w_2\\ w_3\\ ...\\ w_N\end{pmatrix}$$

with the condition $(w_1, w_2, w_3, ...)\begin{pmatrix}1\\ 1\\ ...\\ 1\end{pmatrix} = 1$ or $(w_1, w_2, w_3, ...)\begin{pmatrix}1\\ 1\\ ...\\ 1\end{pmatrix} - 1 = 0$

This is the optimization problem of quadratic convex. Using Lagrange multiplier method, will solve the problem of portfolio optimization which is given by:

$$L(w, \lambda) = 2\tau(w_1, w_2, w_3, ..., w_n)\begin{pmatrix}\mu_1\\ \mu_2\\ ...\\ \mu_N\end{pmatrix} - (w_1, w_2, ..., w_n)\begin{pmatrix}\sigma_1^2 & \sigma_{12} & \dots & \sigma_{1N}\\ \sigma_{21} & \sigma_2^2 & \dots & \sigma_{2N}\\ ... & ... & ... & ...\\ \sigma_{N1} & \sigma_{22} & \dots & \sigma_N^N\end{pmatrix}\begin{pmatrix}w_1\\ w_2\\ w_3\\ ...\\ w_N\end{pmatrix} - \lambda(w_1, w_2, w_3, ..., w_N)\begin{pmatrix}1\\ 1\\ ...\\ 1\end{pmatrix}$$

$-1 = 0$

To solve this Langrange equation, we have to solve two conditions (first-order conditions):

$$L'(w) = 0 \text{ and } L'(\lambda) = 0$$

Completed those conditions, the equation for the optimal porfolio weights as follows:

$$\begin{pmatrix}w_1\\ w_2\\ w_3\\ ...\\ w_n\end{pmatrix} = \frac{1}{(1,1,1,1,...,1)A\begin{bmatrix}1\\ 1\\ 1\\ ...\\ 1\end{bmatrix}}A\begin{bmatrix}1\\ 1\\ 1\\ ...\\ 1\end{bmatrix} + \tau\{A\begin{pmatrix}\mu_1\\ \mu_2\\ ...\\ \mu_n\end{pmatrix} - \frac{(1,1,1,1,1,...,1)A\begin{pmatrix}\mu_1\\ \mu_2\\ ...\\ \mu_n\end{pmatrix}}{(1,1,1,1,1,...,1)A\begin{pmatrix}1\\ 1\\ ...\\ 1\end{pmatrix}}A\begin{pmatrix}1\\ 1\\ ...\\ 1\end{pmatrix}\}$$

Where A was denoted as $A = \Sigma^{-1} = \begin{pmatrix} \sigma_1^2 & \sigma_{12} & \dots & \sigma_{1N} \\ \sigma_{21} & \sigma_2^2 & \dots & \sigma_{2N} \\ \dots & \dots & \dots & \dots \\ \sigma_{N1} & \sigma_{22} & \dots & \sigma_N^N \end{pmatrix}^{-1}$

Now we can calculate the optimal weight vector $w' = (w_1, w_2, ..., w_n)$ by using covariance matrix Σ and matrix $\begin{pmatrix} 1 \\ 1 \\ ... \\ 1 \end{pmatrix}$.

After calculating weight vector w', we can also calculate the variance portfolio return and the expectation of portfolio return τ_p by substituting w' from equation (1) and (2).

With a particular data, we use the process above to determine the portfolio weights. In the weight vector calculation process, the values of risk tolerance τ determined by the simulation value begins 0.000 with an increase of 0.0001. And it is assumed that short sales are not allowed, so we can stop the simulation if there is an entry of w' smaller than 0. After that, we look at the portfolio weight calculation results. For each level of the risk tolerance value, we can calculate the ratio of the mean value of u_p towards the variance. This ratio shows the relationship between the optimum portfolio expected with the variance as a measure of risks. The higher the ration is, the higher the optimum portfolio expected and the lower variance, which means the result is very closed to the mean. Finally, based on the result of the calculation of portfolio optimization, the optimum valued is achieved when the value of ratio is the biggest. So we can conclude weight of the maximum portfolio with corresponding portfolio's risk tolerance τ of that ratio.

To conclude, the entries of the optimal weight vector will make a reference for the investors in each types of stocks, in order to achieve the maximum value of the portfolio [**3**]. To see how this method apply to the real-world situation, we will look at the Application part of this report.

3 Real World Application

We write a Python program based on the aforementioned theory to get hands on experience. Our coding program will allow people to enter their stocks portfolio and help them to calculate the optimized weights of each stock. We use real-time updated data from Tradingview site. The below code block is used to import required library and package:

```
# Initially, import necessary packages as well as download them in the local machine.
from tvDatafeed import TvDatafeed, Interval # This package helps to read the real price data in Tradingview site
import datetime
import pandas as pd
import numpy as np
import statistics as st
tv = TvDatafeed()
```

Next, our program prompts the user to enter the number of stocks, the symbol of the stocks and the exchange where each stock is listed respectively. Then, we get the price data of those stocks and store them into a list:

```
# First of all, prompt the user to input all the stocks he or she wish to invest
n = int(input("Please input the number of stocks: "))
price_list = []
for i in range(n):
    # In order to get the data successfully in Tradingview, for each stock, the user should prompt
    # the three letters ticker symbol of the stock and its exchange like HOSE, HNX, UPCOM
    stock = input("Please input the ticker symbol: ")
    exchange = input("Please input the exchange of stock: ")
    x = tv.get_hist(stock,exchange)
    # We focus on the closing price
    price_list.append(x["close"])
```

Next, we calculate the returns and the average returns of each stock to find the vector of stock return:

```
# Second of all, find the return and variance for each stock
# Return = Pt / Pt-1
return_list = []
variance_list = []
all_return_list = []
for x_stock in price_list:
    x = []
    for i in range(1,len(x_stock)):
        x.append((x_stock[i]/x_stock[i-1])-1)
    return_list.append(st.mean(x))
    variance_list.append(st.variance(x))
    all_return_list.append(x)
```

Next, we compute the covariance matrix:

```
# Third of all, construct the covariance matrix from the first stock to the nth stock
covariance_matrix = np.cov(np.array(all_return_list))
if np.linalg.det(covariance_matrix) != 0:
    inv_cor_matrix = np.linalg.inv(covariance_matrix)
else:
    inv_cor_matrix = np.linalg.pinv(covariance_matrix)
```

Next, we prepare the data - matrices in the right shape - for the equation to find the optimal portfolio:

```
# Prepare data for optimal formula in the model
one_vector_horizontal = np.array([1]*n)
one_vector_vertical = one_vector_horizontal.reshape(n,1)

return_list_vertical = np.array(return_list).reshape(n,1)

result = []
multiply_1 = inv_cor_matrix@one_vector_vertical
determinor = one_vector_horizontal@inv_cor_matrix@one_vector_vertical
multiply_return = inv_cor_matrix@return_list_vertical
multiply_1_1= one_vector_horizontal@inv_cor_matrix@return_list_vertical
```

Then, for each risk tolerance rate, we calculate the weights of each stocks, the expected return and the variance of the portfolio corresponding to those weights as well as its ration (note that short trading is not allowed in Vietnam so we will stop when there is a weight smaller than 0:

```
while(flag):
    weight = (1 / determinor[0]) * multiply_1 + t * (multiply_return - ((multiply_1_1[0] / determinor[0]) * multiply_1))
    for each_weight in weight:
        if each_weight < 0:
            flag = False
            break
    if flag == True:
        t_list.append(t)
        result.append(weight)
        t += 0.001

        # Find the value of the sum column. In theory, the summation on the weight vector should be 1.
        sum_list.append(np.transpose(weight) @ one_vector_vertical)

        # Find the mean of return with the inclusion of the weight
        # Formula: mean = r1.w1 + r2.w2 + ... + rn.wn
        s = 0
        for i in range(len(return_list)):
            s += return_list[i]*weight[i][0]
        mean_list.append(s)

        # Find the variance of return
        variance_list.append(np.transpose(weight) @ covariance_matrix @ weight)
        # Find the maximum. It is equivalent to the subtraction between the mean and the variance
        maximum_list.append(mean_list[-1] - variance_list[-1])
        # Find the ratio. It is equivalent to the subtraction between the mean and the variance
        ratio_list.append(mean_list[-1]/variance_list[-1])
```

Finally, we return a csv file that contains a data frame that contains all information including risk tolerance rate, weights, expected returns, variance, ratio for each combination of the given portfolio as well as print out the information of the optimal portfolio:

```
# In theory, the optimal portfolio is the one having the largest ratio value
biggest_ratio = max(list(ratio_list)) # Find the largest ratio value
# Create a data frame to show all the weight possibles
data  = pd.DataFrame({"t":t_list, "weight": result, "sum": sum_list, "mean": mean_list, "variance": variance_list, 'maximum': maximum_list, "ratio": ratio_list})

# Locate the index of the largest ratio
for i in range(len(ratio_list)):
    if ratio_list[i] == biggest_ratio:
        index = i

# Convert the result to csv file if necessary
data.to_csv('file_name.csv', encoding='utf-8')
# Pring the result which is the optimal portfolio
print(data.iloc[index,:])
```

For example, when we test our program with two stocks GEX and STB listed in HOSE, we receive the below results.

```
Please input the number of stocks: 2
Please input the ticker symbol: GEX
Please input the exchange of stock: HOSE
Please input the ticker symbol: STB
Please input the exchange of stock: HOSE
t                                                                0.072
weight      [[0.00047571801717033457], [0.9995242819828296]]
sum                                                          [[1.0]]
mean                                                        0.001436
variance                                  [[0.0017320612603561492]]
maximum                                  [[-0.0002957227851972557 4]]
ratio                                        [[0.8292654007304286]]
Name: 72, dtype: object
```

So the optimized weights for GEX and STB must be 0.0005 and 0.9995. We can check the result with the data frame in the csv file produced by our program:

	t	weight	sum	mean	variance	maximum	ratio
		[[0.40639163]					
0	0	[0.59360837]]	[[1.]]	-0.00869543	[[0.00100257]]	[[-0.009698]]	[[-8.67310574]]
		[[0.4007539]					
1	0.001	[0.5992461]]	[[1.]]	-0.008554711	[[0.00100271]]	[[-0.00955743]]	[[-8.53155054]]
2	0.002	[[0.39511618]	[[1.]]	-0.008413992	[[0.00100314]]	[[-0.00941713]]	[[-8.38768117]]
		[[0.38947846]					
3	0.003	[0.61052154]]	[[1.]]	-0.008273273	[[0.00100384]]	[[-0.00927711]]	[[-8.24162157]]
4	0.004	[[0.38384074]	[[1.]]	-0.008132554	[[0.00100483]]	[[-0.00913738]]	[[-8.09349905]]
5	0.005	[[0.37820302]	[[1.]]	-0.007991835	[[0.00100609]]	[[-0.00899793]]	[[-7.94344399]]
6	0.006	[[0.3725653]	[[1.]]	-0.007851116	[[0.00100764]]	[[-0.00885878]]	[[-7.7915894]]
7	0.007	[[0.36692758]	[[1.]]	-0.007710397	[[0.00100947]]	[[-0.00871987]]	[[-7.63807057]]
8	0.008	[[0.36128986]	[[1.]]	-0.00756 9678	[[0.00101158]]	[[-0.00858126]]	[[-7.48302467]]
9	0.009	[[0.35565214]	[[1.]]	-0.007428959	[[0.00101397]]	[[-0.00844293]]	[[-7.32659033]]
10	0.01	[[0.35001442]	[[1.]]	-0.00728824	[[0.00101665]]	[[-0.00830489]]	[[-7.16890727]]

67	0.067	[[0.02866432]	[[1.]]	0.000732743	[[0.00163426]]	[[-0.00090152]]	[[0.44836364]]
68	0.068	[[0.0230266]	[[1.]]	0.000873462	[[0.00165326]]	[[-0.0007798]]	[[0.52832778]]
69	0.069	[[0.01738888]	[[1.]]	0.001014181	[[0.00167254]]	[[-0.00065836]]	[[0.60637307]]
70	0.07	[[0.01175116]	[[1.]]	0.0011549	[[0.0016921]]	[[-0.0005372]]	[[0.68252613]]
		[[0.00611344]					
71	0.071	[0.99388656]]	[[1.]]	0.001295619	[[0.00171194]]	[[-0.00041632]]	[[0.75681429]]
		[[4.757180176-04]					
72	0.072	[9.995242824e-01]]	[[1.]]	0.001436138	[[0.00173206]]	[[-0.00029572]]	[[0.8292654]]

4 Limitations and obstacles

4.1 Limitations

Firstly, the changes of market affects some stable variables. The conventional asset allocation model assumes linearity between the portfolio's variables. However, during periods of market changes unstably, the relationship between different types of stocks increase markedly and temporarily differs from long-term correlation levels

Moreover, in real life, the will be changes in the mean-variance method. Because in this method, it was assumed in the mean-variance method that portfolio managers make their decisions of asset allocations once and for all at the beginning of a given period, such as one month or one year, and these assumptions are not allowed to be modified until the end of the period.

Furthermore, this method lacks of subjective measurements. The amount of information (for example, the covariance matrix) needed to compute a mean-variance optimal portfolio is certainly has no room for subjective measurements ('views' about the returns of portfolios of subsets of investable assets or "behaviours" of people).

4.2 Obstacles

Although this is an attempt to provide the current literature with another portfolio optimization method, there are currently more relevant approaches to portfolio optimization, including Least square, Singular Value Decomposition, and our chosen Markowitz. At first, our group struggled in choosing the appropriate method. However, with the help of professor Hung Tran and his explanation of some criteria to select a credifiable reference, we can finalize our idea which is the Markowitz model.

Nevertheless, the second challenge arises when conducting research on this method. The project not only concerns Linear Algebra but also Calculus, Statistics, and Economics which we do not have the opportunity to discuss during

class time. Even though the function to find the result is mainly based on matrix manipulation, the logic behind the algorithms was difficult to grasp.

To resolve the challenge, all members are assigned a part to conduct further research and then report to other team members. After that, we arrange multiple meetings during our project so that all members have a chance to explain their understanding to each others to ensure everyone has a solid understanding in order to be on the same page to write the report and present to the class.

In other words, our project works because everyone is now understanding the meaning of the concept and more importantly the flow to solve for the optimal portfolio. Last but not least, the technical difficulties including advanced Latex and Python is also a matter to our group. With latex, it is quite difficult to structure a mathematical report as it is the first time for all of us to write a LaTex report. On the other hand, with Python, the job is tough because Google Colaboratory does not offer some packages to get the data, so we have to install all of them to our local machine which takes a lot of time to set up the environment.

Nonetheless, the problem is solved easily with Pair-programming. We work together on the same task to observe and search google to fix quickly. As a result, on the one hand, this project consists of a number of obstacles both logistically and technically. On the other hand, resolving these issues is a process of active learning and improvement.

5 Authors' Contributions

1. Workload of each contributors

The workload of this project was delegated to the researchers according to this table

Performing	Finance knowledge and terminologies	All members
	Linear algebra knowledge	
	Other mathematics functions (variance, Lagrange multiplier function, etc.)	
	Review existing solutions	Khanh Quan, Doan Thuan
	Code	
Written report	Build content	Ngoc Ha, Phuong Nam, Tien Dung
	LaTex format	
Presentation	Build content	All members
	Design slide	

2. Details timeline

From 11/21/2021-12/2/2021: Forming team and ideation: To begin with the project, we have formed a team consisting of 5 members with different background and skillset including mathematics, programming, economics, and academic writing in order to complement each others.

From 12/1/2021-12/15/2021: Planning idea: As our group consists of a diverse background, the idea comes widely from either from the suggestion of the instructor or other Linear Algebra references. As a short list, the ideas are SVD image compression, Machine Learning on voice recognition as well as prediction. However, we think that these ideas may not be so interesting to the subject and can be also duplicated with other groups. Therefore, we spend more time on planning on the new pathway. As a quick observation, there are some Economic-major members in the group, so why do

not we change our direction to stock investment. As a result, the idea of portfolio optimization arises which we believe will become an interesting project for the class.

From 12/15/2021 to 12/22/2021: Finding appropriate approach: As the project idea is interesting, there are a lot of methods to solve. For doing research, we find that Least square, Singular Value Decomposition, and Markovitz may be suitable to our demand. Thanks to the kind support from professor Hung Tran, we can end up following the Markovitz model.

From 12/22/2021 to 12/29/2021: Research focusing on Markovitz model: The model stated in our reference document requires some knowledge beyond Linear Algebra such as Calculus, Statistics, and Economics. Because of that, our group divides the materials equally to our group members to do further research. After that, we schedule a meeting on weekend for a deep discussion to ensure that all members understand what we are doing.

From 12/29/2021 to 1/6/2022: Implementing the solution on Python: After getting the idea of Markovitz model, we decide to simulate it on Python in order to be a program that the users can prompt n pre-chosen securities as the input with the output is the weight vector that the investors should distribute to their portfolio.

From 1/6/2022 to 1/13/2022: Writing report: The next step is to assemble all of the deliverables from previous weeks to write a holistic report followed with the criteria given by the instructor. In this step, we have one to two meetings to work out the clear structure of the report and to check in the progress among all members.

From 1/14/2022 to 1/21/2022: Report revision and final presentation: The final week of this project is dedicated to revise the work on the report carefully. Then, we also meet to select the suitable part to put in the actual presentation in order to attract the audience.

References

[1] B. Basuki, S. Sukono, D. Sofyan, S. S. Madio, and N. Puspitasari *Linear Algebra on Investment Portfolio Optimization Model, Journal of Physics: Conference Series, vol. 1402, no. 7, p. 077089, 2019.*

[2] Hass, Joel, and Maurice D. Weir. *Thomas' Calculus: Early Transcendentals.*

[3] S. Stojanovic *Investment portfolio optimization, Neutral and Indifference Portfolio Pricing, Hedging and Investing, 2011.*