Asset allocation optimization based on linear and quadratic programming models

Larry Cao

Harrow School, London, United Kingdom

Abstract. The aim of this paper is to provide a detailed insight into two mathematical models, one linear and one non-linear, that tackles the asset allocation optimization problem. We have collected data for seven different investment options in the last decade. The data are then analyzed with python, including visualizing them with several different graphs and computing their covariance and means. Our models are solved by python as well, admitting two different asset allocation plans according to application scenarios.

Keywords: Linear programming, portfolio optimization, covariance, python, investment options.

1. Introduction

In the current society that we live in, having the ability to accurately assess the financial market and invest can bring with it fruitful financial gains to an individual or a corporation. A portfolio is an investment that is managed by professional investors or a professional institution, whereas an optimal portfolio is the investment option that brings about the most profit within the investor's risk tolerance. [2]

Up to 2022, many different choices of investment have become available as money making opportunities. The following introduces some mostly considered options for many families or individuals. Firstly, investing in equity funds is an option, however, it is a risky one, as this fund deals primarily with stocks, which is highly unpredictable. The second option is investing in commingled funds. This option is slightly less risky than equity funds, but much of commingled funds still relies on stocks to profit. The third option is bond funds. This choice is undoubtedly a less risky choice, and from the graph below it is easy to see that this type of investment option provides positive profit every year. Next, gold is another investment option that I am going to bring into my calculations. Gold is an investment choice that varies depending on the political state of the world, since its price is inversely proportional to that of the dollar. After that, we have the choice of investing in commodities/goods. This type of investment is also a risky option, as the value of goods depends on many factors. Real estate is another option for investment, however, in recent years new government policies have restrained the profitability of this choice. Finally, WIP is one of the safest options possible, but along with this benefit brings the downside that little profit can be made.

This paper focuses on the discussion of two different mathematical models for optimizing. The first one is a linear model which finds the best solution through pre-defined limits. This model is suitable for experienced families or individuals. The second model is a non-linear model with the addition of the concept of risk. In this model, risk is represented by the covariance between the different investment options. In fact, the basis of this model was first introduced in 1952 by Markowitz [1]. The Markowitz's portfolio optimization model defines the risks of an investment option as the magnitude of the data dispersion, as well as the amount of influence different branches of that data (e.g between different stocks) have on each other.The model utilizes a weight coefficient between 1 and 0 to represent the relative importance of risk and benefits of the investment choice. This ground-breaking thesis helped Markowitz obtain the Nobel prize for economics in 1990.

After collecting real data for the aforementioned seven different investment options in the last decade, we use python to do data analysis and data visualization, mainly by the Pandas and Seaborn python libraries. Then, we solve our two models by python as well, with the pulp and cvxopt python packages respectively. As a result, we have obtained two separate asset allocation solutions. The detailed graph containing all the profit rates will be shown further down.

The rest of this paper is organized as follows. In Section 2, we give analysis of the different ways of visualizing data. In Section 3, we discuss the basics of the formulation of the linear model, as well as obtain its results. In Section4, we will look at the formulation of the non-linear model and the meaning of its results. Conclusions are given in Section 5.

2. Data for the research and data analysis

In this section, we discuss the different ways of data visualization used in this paper.

The table 1 shows the profit rates collected in the past decade for seven different investment options:

Table 1 Profit margin of seven different investment options in the past ten years

	Equity Fund	Commingled Funds	Bond Fund	Gold	Goods	Real Estate	WIP
2012	4.90%	3.78%	7.04%	7.10%	4.20%	-0.10%	5.30%
2013	14.42%	12.22%	0.98%	-28%	-12.40%	9.20%	5.10%
2014	28.93%	18.61%	17.82%	-1.80%	-16.50%	-4.30%	5.70%
2015	31.23%	36.60%	10.77%	-10%	-14.50%	0.20%	5.10%
2016	-9.14%	-7.80%	0.39%	8.50%	51.30%	10.50%	4.10%
2017	12.59%	10.11%	2.01%	13.20%	7.90%	5.60%	4.60%
2018	-25.09%	-13.59%	4.25%	-1.60%	-5.80%	5.10%	4.80%
2019	35.97%	29.48%	5.91%	18.30%	15.60%	3.30%	4.30%
2020	45.94%	47.47%	4.02%	25.80%	-6.10%	3.10%	4%
2021	9.98%	8.69%	4.91%	-3.65%	30.49%	2.44%	4.40%

We have stored the data in an excel table named “portfolio_data.xlsx”, which will be used later in the python code for data visualization, as well as solving the models.

The first graph（Fig.1） is a typical line graph, representing the movement of the financial profits made from the different options in the last 10 years.

Fig.1 Line graph representing the profit rate of different investment options

The second graph（Fig.2） is a boxplot, indicating the dispersion and the skewness of the data. It also demonstrates some important values such as the interquartile range of the data.

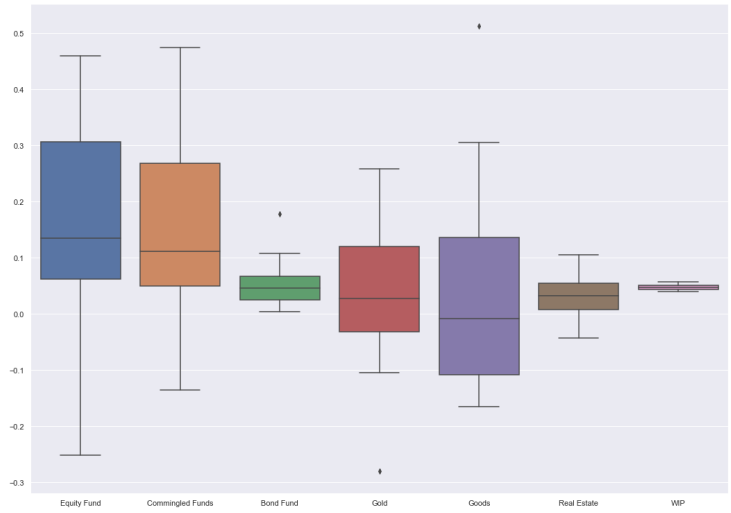

Fig.2 A boxplot for dispersion and skewness

The final graph（Fig.3） is a heatmap. The heatmap depicts the covariance of any two strands of data in the form of a matrix.

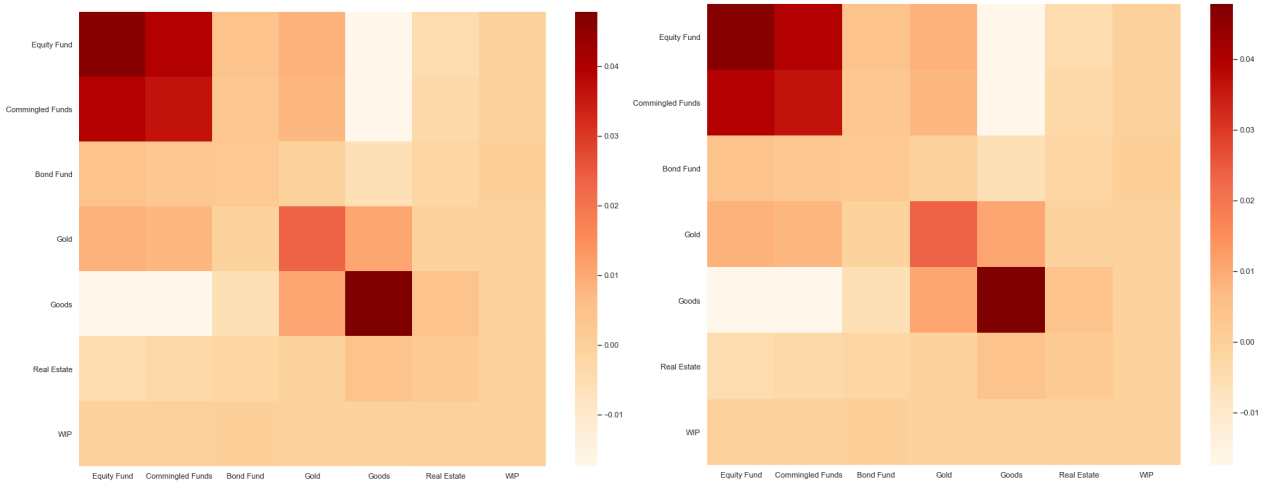

Fig.3 Heatmap for covariance between two options

The definition of covariance of two random variables is as follows:

Definition 1. Let x, y be two discrete random variables with means $\bar{x}$, $\bar{y}$ respectively. The covariance of them is defined as

$$cov_{x,y} = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{N}.$$

Covariance is a measure of the joint variability of two random variables. If the two variables have a positive correlation (if a greater value of the first variable corresponds with a greater value of the second, and if the same holds for lesser values), the covariance is positive.

3. Linear programming model for asset allocation

In this section we give the method for the linear model, the explanation for the constraints added, as well as its result.

Let w_1, w_2, w_3, w_4, w_5, w_6, w_7 be the percentage of how much money should be put into the different investment options. Let r_1, r_2, r_3, r_4, r_5, r_6, r_7 be the average profit rate of a particular choice in the last decade.

Below is the linear programming model for our asset allocation problem:

Max $\sum_{i=0}^{7} w_i r_i$

s.t.

$w_1 + w_2 + w_3 + w_4 + w_5 + w_6 + w_7 = 1$

$w_1 \leq 0.08$

$w_1 + w_2 \leq 0.20$

$w_3 + w_7 \geq 0.40$

$0.05 \leq w_4 \leq 0.08$

$w_5 \geq 0.08$

In the above model, the objective function represents the expected profit made.

Explanation of constraints:

All the percentages must add up to 1, because we assume that all the money is invested into the options.

Equity fund is a very risky approach, so we would not invest more than 5% into this option.

Commingled funds is largely dependent on stocks, so I would not invest a total of more than 15% into equity funds and commingled funds combined.

Being a conservative investor, I would like to invest primarily in bond funds and WIP (e.g storing my money in banks). I would like to invest a total of more than 50% into these two options.

The profitability of gold varies with the political state of the world. Therefore, depending on the profit made for the last decade, I would like to invest no less than 5% but no more than 8% into gold.

Real estate is normally a very profitable industry in China, however, recent policies have attempted to reduce the potential money existent in this investment option. Despite of the policies, it is inevitable that in many places (such as houses near schools), real estate is still very profitable. I would like to invest a minimum of 8% in real estate.

The python code uses the portfolio_data excel sheet and the PuLP optimization library to work out the optimized solution. The python code for the model is shown in the appendix. The result is shown in the image below（Fig.4）:

```
w1: 0.08
w2: 0.12
w3: 0.4
w4: 0.05
w5: 0.0
w6: 0.05
w7: 0.3
Objective value: 0.07003129999999999
```

Fig.4 Python solution result

The results above demonstrate the ratio of money that should be invested into each of the different choices. Some interesting observations can be made from the percentages. The linear model contains multiple constraints. They are certainly not fixed, which means depending on the data collected from the last decade of the investment option, its limitations should also be altered accordingly. For example, we currently are setting the lowest bar on investing in real estate at 5%. However, if in future years policies start kicking in and the profits drop, I will most likely lower the limitation. Similarly, if the stock market encounters difficulties, w1 can be lowered even further from 8%. The linear model outputs a relatively high anticipated income percentage. That is because it does not take in the element of risk in proper sophistication. It is sometimes not enough to simply rely on the mean value to determine calculations in the model.

Using the constraints determined above, the maximum profit we can make is 7.00% (to 3 significant figures)

4. Quadratic programming model for asset allocation

In the non-linear model, the concept of matrix is used to solve the optimal investment ratio. A weight coefficient (mu) is defined as a variable in this model. The purpose of the weight coefficient is to balance the outcome of the solution due to the relative importance of the profit and risk to the user of this model.

This model is created by incorporating new elements of mathematical modelling such as matrices and the concept of covariance, which is defined earlier in the code by the line cov_mat=df.cov(). The relative covariances of different investment options on each other are visually represented by the heatmap above.

The non-linear model relies on a weight variable. In this model, μ (the weight variable) gives out the maximum profit between the range 0.028 and 0.14. When mu takes values outside of this range, w1…7 may come out as negative, which are not valid solutions. Covariance is applied in this model, which explains why the objective value is smaller. The model is also more sophisticated in the sense that the percentages are given in deeper detail (e.g 5.257992402351851e-08) as opposed to being rounded to zero. On the other hand, pre-defined constrictions are lost, which is undoubtedly a downside of this model. The objective function is as follows, where w represents the percentage of money invested into a certain option, and σ represents the covariance. μ is the input variable we choose in this model to represent the risk tolerance:

$\min \sum_{i,j} w_i w_j \sigma_{i,j}$

s.t.

$$\sum w_i = 1$$

$$\sum_i w_i \mu_i = \mu$$

Non-linear programming using matrices and the μ variable gives the following percentages as the optimized result. The income percentage is also given. This is the optimized result when the weight coefficient μ is set to 0.14（Fig.5）.

```
     pcost       dcost       gap    pres   dres
 0:  1.0502e-02 -8.2118e+00  8e+00  6e-16  4e-16
 1:  1.0464e-02 -1.1272e-01  1e-01  2e-16  1e-15
 2:  9.2520e-03 -3.1853e-03  1e-02  1e-16  9e-17
 3:  8.1050e-03  6.2755e-03  2e-03  2e-16  2e-17
 4:  7.6775e-03  7.5858e-03  9e-05  1e-16  1e-17
 5:  7.6574e-03  7.6564e-03  1e-06  2e-16  8e-18
 6:  7.6572e-03  7.6572e-03  1e-08  2e-16  1e-17
Optimal solution found.
------------the optimal solution is listed below:--------------------------------
w1:  7.873752516679943e-08
w2:  0.4889638187570504
w3:  0.2886170856466693
w4:  2.8887793570377068e-08
w5:  0.22241759978905232
w6:  3.145037316003758e-07
w7:  1.0736781776036773e-06
objective value:  0.007657186647450239
```

Fig. 5 Optimization results

In the following graph（Fig.6）, we have gone through μ values to illustrate a border, in which all other possible μ values reside in.

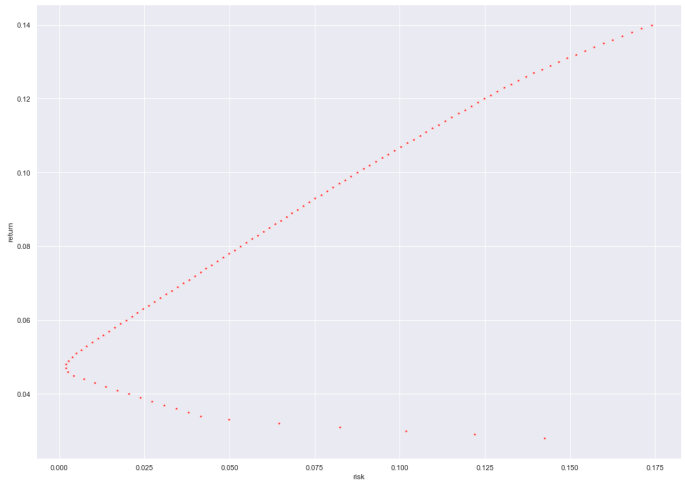

Fig. 6 Boundary value

In the graph above the x-axis represents the risk the μ value brings, while the y-axis represents the maximum profit possible at a particular risk level.

We have chosen 10000 μ values at random to plug into the model, and the results are shown in the following graph（Fig.7）:

Fig. 7 Random Value Results

5. Conclusion

Knowing how to optimize the allocation of assets is becoming increasingly significant. The paper presented above provides two distinct models that aim to provide the best solution for the portfolio optimization problem. The two models given contained one linear model and one non-linear. We have discussed the meaning of the results obtained, and why they differ from each other. The linear model has the downside of not measuring the level of risk, while the non-linear model does not contain the feature that allows limits to be added beforehand. The models and the results of the investigation are applicable to real-life data. They can be helpful to individuals or large corporations looking to invest. [3]

Appendix

1. Python code for data analysis and visualization

```
df = pd.read_excel("./portfolio_data.xlsx", header=0, index_col=0)
```

```
profit = df.values.transpose()

sns.set(rc={"figure.figsize":(18, 13)})
sns.lineplot(data=df)
plt.show()

sns.boxplot(data=df)
plt.show()

expt_returns = df.mean()
cov_mat = df.cov()

sns.set(rc={"figure.figsize":(15, 12)})
sns.catplot(data=df, kind=bar)
plt.show()

sns.heatmap(cov_mat, cmap=OrRd)
plt.show()
```

2. Python code for solving linear programming model

```
from pulp import *


# Create a new model
m = LpProblem(name="MIP_Model", sense=LpMaximize)

# Create variables
w1 = LpVariable("w1", 0, 1)
w2 = LpVariable("w2", 0, 1)
w3 = LpVariable("w3", 0, 1)
w4 = LpVariable("w4", 0, 1)
w5 = LpVariable("w5", 0, 1)
w6 = LpVariable("w6", 0, 1)
w7 = LpVariable("w7", 0.4, 1)

# Set objective function
m += expt_returns.values[0]*w1 + expt_returns.values[1]*w2 + \
      expt_returns.values[2]* w3 + expt_returns.values[3]*w4 + \
      expt_returns.values[4]*w5 + expt_returns.values[5]*w6 + \
      expt_returns.values[6]*w7, obj

# Add constraints
m += w1 + w2 + w3 + w4 + w5 + w6 + w7 == 1, c1
```

```
m += w1 <= 0.05, c2
m += w1 + w2 <= 0.15, c3
m += w3 + w7 >= 0.5, c4
m += w4 >= 0.03, c5
m += w4 <= 0.05, c6

# Calculate with the default CBC optimizer
status = m.solve()

if LpStatus[status] == Optimal:
    for v in m.variables():
        print(f{v.name}: {v.varValue})

print(fObjective  value: {m.objective.value()})
```

3. Python code for solving the non-linear model

```
from cvxopt import matrix, solvers


n_asserts = 7
mu = .1
P = 2 * matrix(cov_mat.values)
q = matrix(np.zeros(n_asserts))
G1 = -matrix(np.eye(n_asserts))
h1 = matrix(0.0, (n_asserts, 1))
G2 = matrix(np.eye(n_asserts))
h2 = matrix(1.0, (n_asserts, 1))
G = matrix([G1, G2])
h = matrix([h1, h2])
A1 = matrix(1.0, (1, n_asserts))
b1 = matrix(1.0)
A2 = matrix(expt_returns.values, (1, n_asserts))
b2 = matrix(mu)
A = matrix([A1, A2])
b = matrix([b1, b2])

sol = solvers.qp(P, q, G, h, A, b, verbose=False)
print( ----------the optimal solution is listed below:----------)
for i in range(len(sol[x])):
    print(fw{i+1}: , sol[x][i])

print(objective value: ,  sol[primal objective])
```

4. Python code for displaying the curve for all possible mu values

```
from cvxopt import matrix, solvers
from math import sqrt


def portfolio(mu):
    n_asserts = 7
    P = 2 * matrix(cov_mat.values)
    q = matrix(np.zeros(n_asserts))
    G1 = -matrix(np.eye(n_asserts))
    h1 = matrix(0.0, (n_asserts, 1))
    G2 = matrix(np.eye(n_asserts))
    h2 = matrix(1.0, (n_asserts, 1))
    G = matrix([G1, G2])
    h = matrix([h1, h2])
    A1 = matrix(1.0, (1, n_asserts))
    b1 = matrix(1.0)
    A2 = matrix(expt_returns.values, (1, n_asserts))
    b2 = matrix(mu)
    A = matrix([A1, A2])
    b = matrix([b1, b2])
    return solvers.qp(P, q, G, h, A, b)[primal objective]

mus = [i/1000 for i in range(28, 141)]
sols = [portfolio(mu) for mu in mus]
risks = [sqrt(sol) for sol in sols]

data = pd.DataFrame({risk:risks, return:mus})
sns.set(rc={"figure.figsize":(18, 13)})
sns.scatterplot(x=risk, y=return, data=data, marker=*, color=Red)
plt.show()
```

5. Python code for demonstrating random points on the profit-risk graph

```
def rand_weight(n):
    k = np.random.rand(n)
    return k / sum(k)


p = np.asmatrix(expt_returns.values)
C = np.asmatrix(cov_mat.values)


n_points = 10000
```

```
rand_returns = []
rand_risks = []
rand_sharpes = []
while n_points >0:
    w = np.asmatrix(rand_weight(n_asserts))
    mu = w * p.T
    sigma = np.sqrt(w * C * w.T)
    rand_returns.append(mu)
    rand_risks.append(sigma)
    rand_sharpes.append(mu / sigma)
    n_points -= 1


rand_risks = [r.getA()[0][0] for r in rand_risks]
rand_returns = [r.getA()[0][0] for r in rand_returns]
rand_sharpes = [r.getA()[0][0] for r in rand_sharpes]

rand_data = pd.DataFrame({risks:rand_risks, expected returns:rand_returns, sharpe ratio:rand_sharpes})
sns.set(rc={"figure.figsize":(18, 13)})
sns.scatterplot(x=risk, y=return, data=data, marker=*, color=Red)
sns.scatterplot(x=risks, y=expected returns, data=rand_data, hue=sharpe ratio, size=sharpe ratio)
plt.xlabel(risks)
plt.ylabel(expected returns)
plt.show()
```

References

[1] H. Markowitz, “Portfolio selection,” Journal of Finance, vol. 7,pp. 77–91, 1952.

[2] N. K. Oladejo, A. Abolarinwa, S. O. Salawu, "Linear Programming and Its Application Techniques in Optimizing Portfolio Selection of a Firm", Journal of Applied Mathematics, vol. 2020, 7 pages, 2020.

[3] Libo, Sun, "Empirical Study of Markowitz's Portfolio Theory Based on Python _ Sun Libo", Times Finance, pg.46 to pg.50