線性代數期末報告—Matrices and CAPM

109071015 計財 24 張晁維

一、動機

幾乎所有的投資都具有一定程度的風險，而這也是為何被投資方須要以利息、股息作為投資者借出資金的回報—用以補償投資者的資金隨時間的貶值以及補償投資人特定時刻區間所承擔的風險。因此便有了俗諺「承擔越多風險，獲得的報酬即越高」。然而我們知道在現實世界各類學派百家爭鳴，這是因為世上沒有人能駕馭風險和報酬，或是在這兩者之間取得完美的平衡。

而一切的不完美使我欲以線性代數和財務管理的知識分析、比較此二者，以下均以水泥類股作為範例，是因為我對這個代碼排在最前端的類股始終感到好奇，因此有了以下的報告欲一窺此一少為投資人討論的產業。

二、目的

(一)以報酬-風險圖分析近十年(2011~2021)的水泥類股

(二)探討相同權重的投資組合與效率前緣的關係

(三)繪製效率前緣

三、本文

這份報告所使用的工具—報酬-風險圖的橫軸為風險、縱軸為平均報酬，由於直接以特定類股為單位分析，因此令特定類股為數支股票的投資組合，圖上每個資料點所對應兩個座標軸的點分別代表風險(變異數、共變異數)和回報(加權平均)，每張圖各有 100,000 個隨機的樣本點、每個樣本點的個股加權合為 1(100%)，同時計算樣本點之間的相關性。

(一)資料

本報告所採用的資料來源於 yahoo finance，以 python 的 yfinance 套件引入。

(二)平均報酬計算方式

使用價格資料中的收盤價作為參考價格，先計算出每日報酬(第 n 日收盤價減第 n-1 日收盤價，再除以每第 n-1 日的收盤價)，在取平均、得平均每日報酬，最後再乘以當年度交易天數、得平均年度報酬。

(三)投資組合風險(變異數、共變異數)計算方式

以收盤價回報以 pandas 套件的功能.cov()取得特定類股中各股之間的變異數與共變異數，並獲得下列矩陣。

$$A = \begin{pmatrix} \sigma 1^2 & \cdots & cov(1,n) \\ \vdots & \ddots & \vdots \\ cov(1,n) & \ldots & \sigma n^2 \end{pmatrix}$$

投資組合的變異數為以上矩陣中每個元素乘以權重、在取所有元素之合。

$A \in R^{n \times n}$

For every element, A_{ij}, if i=j, then the weight of it is $X_iX_j=X_i^2$.

If $i \neq j$, then the weight of A_{ij} is X_iX_j

加權後的矩陣如下。

$$A' = \begin{pmatrix} X1^2\sigma 1^2 & \cdots & X1Xncov(1,n) \\ \vdots & \ddots & \vdots \\ X1Xncov(1,n) & \cdots & Xn^2\sigma n^2 \end{pmatrix}$$

投資組合變異數(風險): $\sigma p^2 = \sum_{i=1,j=1}^{n} Xi * Xj * cov(i,j)$ (if $i = j, \mathrm{cov}(i,j) = \sigma i^2$)

(四)投資組合報酬計算方式

$$B = [X1 \quad \ldots \quad Xn][E(R1) \quad \ldots \quad E(Rn)]^T$$

投資組合報酬: $E(Rp) = \sum_{i}^{n} Xi * E(Ri)$

四、程式介紹

(一)套件

Pandas 用於承載資料與部分計算；yfinance 用於資料收集；numpy、scipy、functools 用於計算、matplotlib 用於繪圖。

```
import pandas as pd
import yfinance as yf
import numpy as np
import matplotlib.pyplot as plt
from functools import reduce
```

(二)資料處理與計算

每日回報取計算結果小數點後 7 位；去除空值的目的為計算報酬時，資料列的第一天沒有報酬因此該值為 NaN；圖中參數為範例僅供參考。其餘資料處理與計算方式如上文。

```
stocks=['1101.tw', '1102.tw', '1103.tw', '1104.tw', '1108.tw', '1109.tw', '1110.tw', '1101B.tw']#股票代碼
expected_return_i=[]#平均回報
stocks_return=np.nan#個股回報(日)
for i in range(0, len(stocks)):#擷取資料
    if i==0:
        n=yf.download(stocks[i], start='2020-01-01', end='2021-01-01')
        stocks_return=round(n['Close'].diff()/n['Close'], 7)
        n5=len(n['Close'])
    else:
        n=yf.download(stocks[i], start='2020-01-01', end='2021-01-01')
        na=round(n['Close'].diff()/n['Close'], 7)
        stocks_return=pd.concat([stocks_return, na], axis=1)
stocks_return.columns=stocks#增加各股回報名稱
stocks_return=stocks_return.dropna()#去除第一位空值
for i in range(0, len(stocks)):#平均回報(日)
    na=stocks_return.iloc[:, i].mean()
    expected_return_i.append(na)
expected_return_i=pd.Series(expected_return_i)#平均回報(年)
covariance_table=stocks_return.cov()*(n5-1)#取變異數、共變異數(年)
```

(三)平均權重的投資組合計算

將特定類別中所有個股取相同、合為 1 的權重，用於與亂數樣本點比較其風險與報酬。

```
#相同權重下的投資組合回報與投資組合風險
weight_avg=np.array([(1/len(stocks))]*len(stocks))#個股權重相同
portfolio_return_avg=sum(expected_return_i*weight_avg)#portfolio  return
portfolio_risk_avg=np.sqrt(reduce(np.dot, [weight_avg, covariance_table,  weight_avg.T]))#portfolio  risk
```

(四)亂數樣本點的投資組合計算

以亂數產生大量樣本點(投資組合)的目的在於標定出概略的可行解在風險-報酬圖上範圍，同時用這些樣本點計算風險與報酬的相關性，作為不同股票類別之間的比較依據。

```
#模擬可行的投資組合在回報-風險圖上分布
portfolio_risk_i, portfolio_return_i=[],  []#用於記錄各點
N=100000#樣本數目
for _ in range(0, N):
    weight=np.random.rand(len(stocks))
    weight=weight/sum(weight)#使各股權重合=1
    return_i=sum(expected_return_i*weight)
    risk_i=np.sqrt(reduce(np.dot, [weight, covariance_table, weight.T]))
    portfolio_risk_i.append(risk_i)
    portfolio_return_i.append(return_i)
```

(五)繪圖

平均權重的樣本點為紅色、亂數產生的樣本點則為藍色。程式結束時會同時輸出前者的風險、報酬，以及後者在風險、報酬之間的相關性。

```
#繪圖
fig = plt.figure(figsize = (10,6))
fig.suptitle('random  simulation', fontsize=20, fontweight='bold')
pic=fig.add_subplot()
pic.plot(portfolio_risk_i, portfolio_return_i, 'o', color='b')
pic.plot(portfolio_risk_avg, portfolio_return_avg,  'o', color='r')
pic.set_title(f'N={N}', fontsize=20)
pic.grid()
fig.show()
print('平均權重報酬', portfolio_return_avg, '平均權重風險', portfolio_risk_avg)
print('亂數點相關係數', np.corrcoef(portfolio_risk_i, portfolio_return_i)[0][1])
```

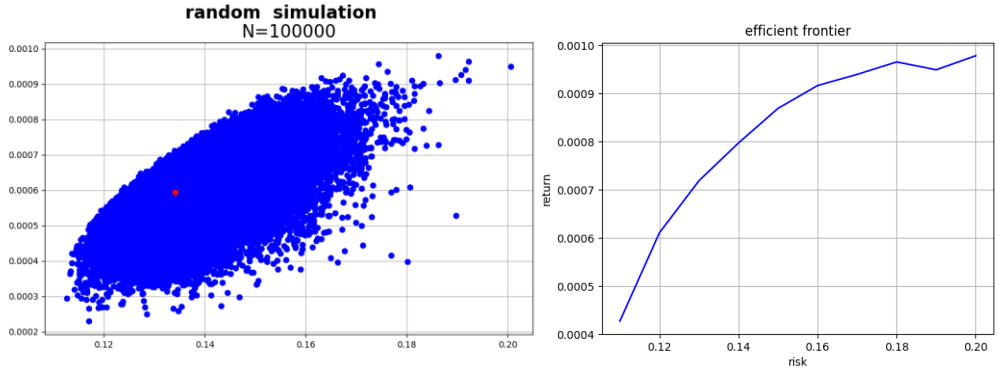
效率前緣的做法則是先將上述的亂數樣本點取至小數點後二位、挑出每個區間中回報最高者，最後再將所有點連成線。

```
#繪圖
print(max(portfolio_risk_i), min(portfolio_risk_i))
for i in range(0, len(portfolio_risk_i)):
    portfolio_risk_i[i]=round(portfolio_risk_i[i], 2)
    portfolio_return_i[i]=round(portfolio_return_i[i], 7)
list_x, list_y=[], []
for i in range(0, len(portfolio_risk_i)):
    if portfolio_risk_i[i] not in list_x:
        list_x.append(portfolio_risk_i[i])
        list_y.append(portfolio_return_i[i])
    else:
        na=list_x.index(portfolio_risk_i[i])
        if list_y[na] >=  portfolio_return_i[i]:
            pass
        else:
            list_y[na]=portfolio_return_i[i]
df=pd.DataFrame({'x':list_x, 'y':list_y})
df=df.sort_values(by='x')
list_x, list_y=df.iloc[:, 0].to_list(), df.iloc[:, 1].to_list()
plt.plot(list_x, list_y, color='b')
plt.title('efficient frontier')
plt.xlabel('risk')
plt.ylabel('return')
plt.grid()
print('平均權重報酬', portfolio_return_avg, '平均權重風險', portfolio_risk_avg)
print('相關係數', np.corrcoef(portfolio_risk_i, portfolio_return_i)[0][1])
plt.show()
```

五、程式運行結果

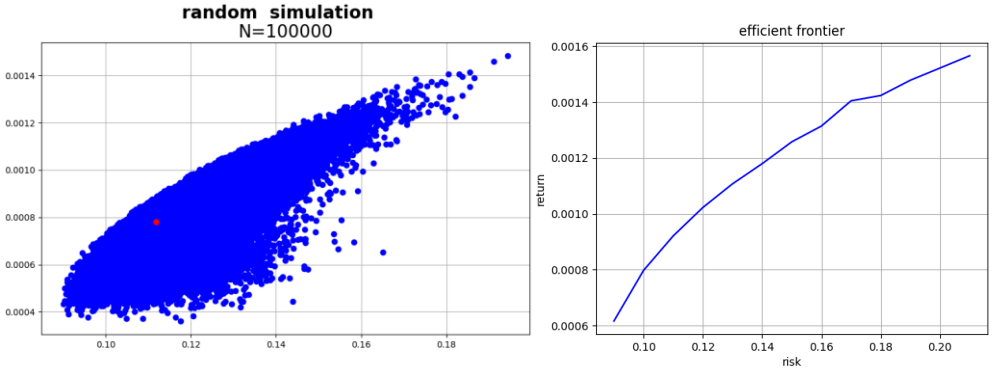
以下左、右圖分別為近十年(2012~2021)的水泥類股的可行投資組合模擬與其效率前緣。

1. 水泥類股(2012)



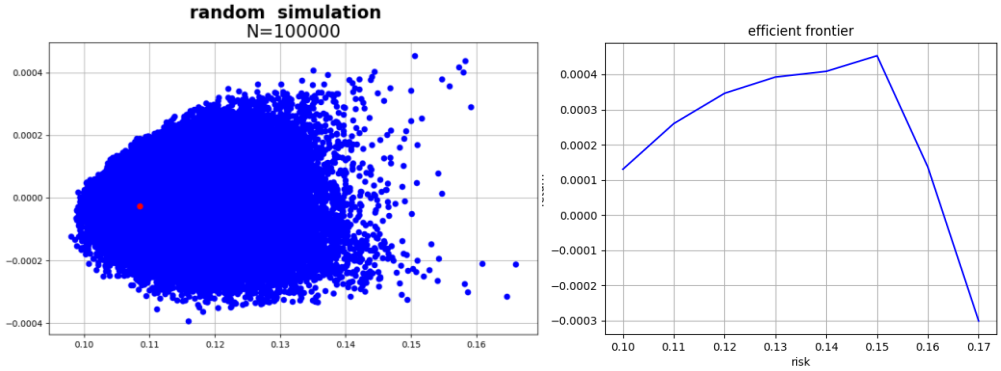
平均權重報酬 0.0005930084441873914 平均權重風險 0.13415887707481322
相關係數 0.664479201588828

2. 水泥類股(2013)



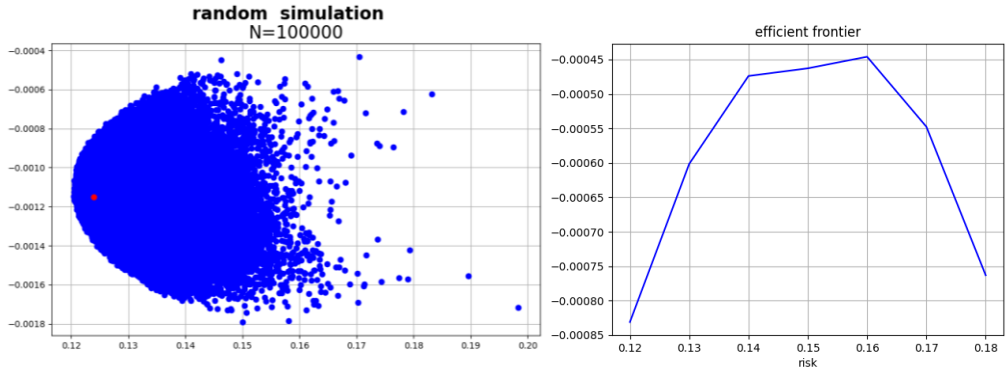
平均權重報酬 0.0007816272192827747 平均權重風險 0.11177062513589932
相關係數 0.7869581167028763

3. 水泥類股(2014)



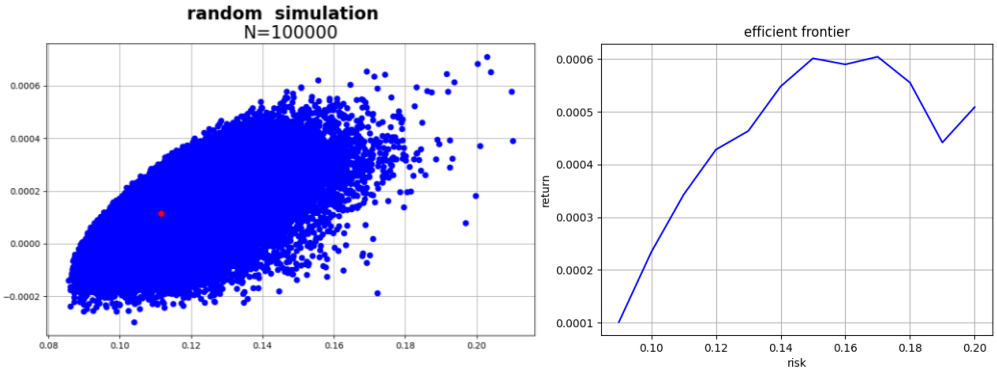
平均權重報酬 -2.5528513591671474e-05 平均權重風險 0.10848340116539688
相關係數 0.01801344668553371

4. 水泥類股(2015)



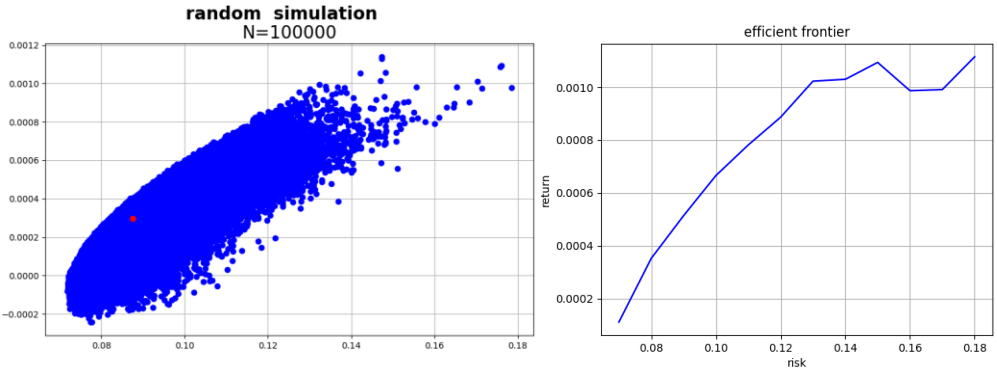
平均權重報酬 -0.0011500654810495626 平均權重風險 0.123912952974410 34 相關係數 -0.16065680119333986

5. 水泥類股(2016)



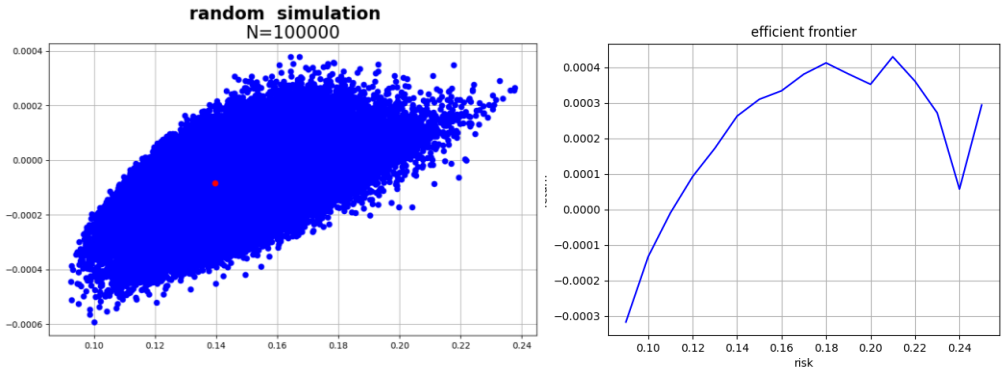
平均權重報酬 0.00011486325690770133 平均權重風險 0.11167432501318487 相關係數 0.6036591350805794

6. 水泥類股(2017)



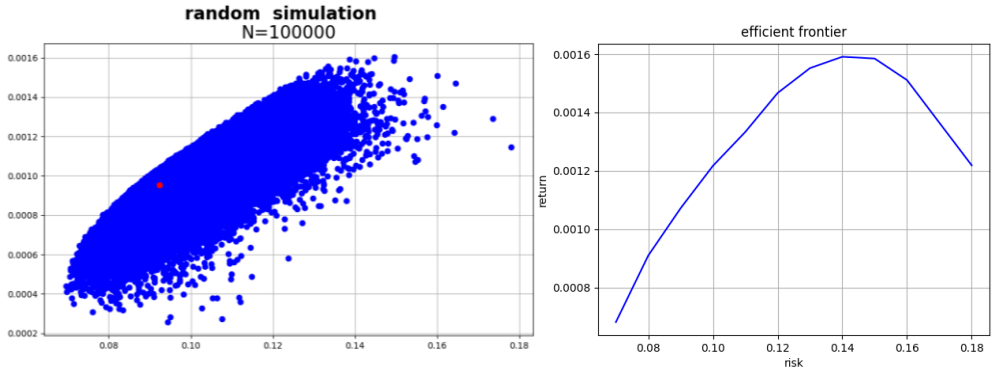
平均權重報酬 0.0002960155844155843 平均權重風險 0.08761523502496858 相關係數 0.801339770291677

7. 水泥類股(2018)



平均權重報酬 -8.276399297423882e-05 平均權重風險 0.13973246995272162
相關係數 0.5330056212653327

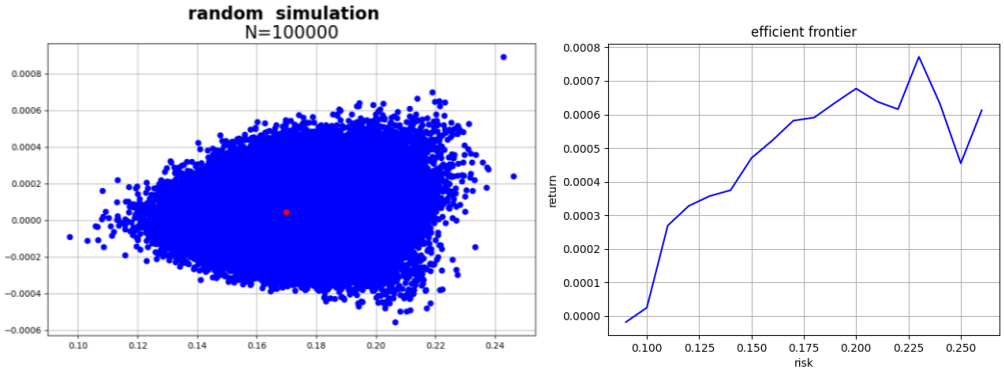
8. 水泥類股(2019)



平均權重報酬 0.0009543432738095238 平均權重風險 0.09246436894361498
相關係數 0.7959727296904613

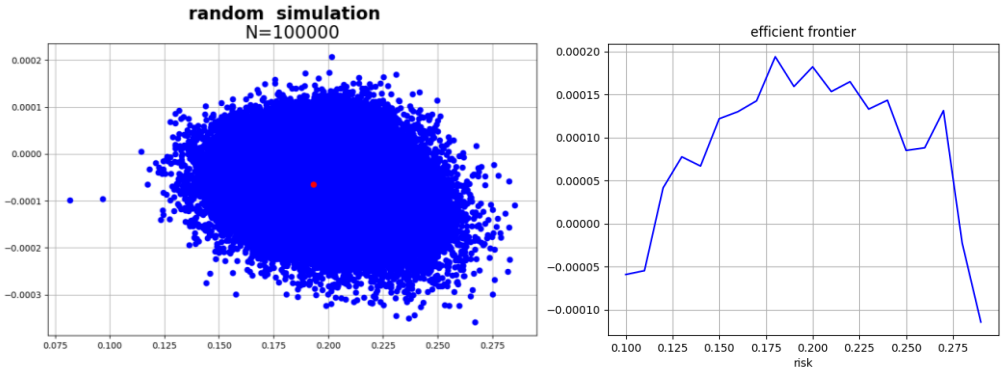
※台泥乙特(1101B.tw)於 2019.1.29 上市，因此自 2020 始採計。

9. 水泥類股(2020)



平均權重報酬 4.526552254098365e-05 平均權重風險 0.1699377786435868
相關係數 0.13395229756633076

10. 水泥類股(2021)



平均權重報酬 -6.479070247933899e-05 平均權重風險 0.19328991457114816
相關係數 -0.22421886267031005

六、結論

根據程式運行結果，顯而易見地，可以觀察到以下現象:

1. 上市的水泥類股具有明顯的產業週期，這個特徵會以可行解(藍色亂數點)分布的差異呈現。
2. 若考慮每年約 2%的通貨膨脹，以相同加權投資水泥類股中各股是虧損的(最高報酬約 0.1%，2019 年)
3. 若報酬與風險相關性越高、亂數點愈集中，則相同加權的投資組合與效率前緣的垂直距離越短(給定風險的報酬愈有效率)；反之則遠離效率前緣。
4. 若風險與報酬相關性越高、亂數點愈集中，則高風險區的報酬愈高、較少下降。
5. 就亂數結果顯示，若不計風險，最高的報酬略低於 0.16%(2012 年)。換句話說，若是維持每年 2%的通貨膨脹，即使投資人承擔很大的風險，亦無法避免虧損，因此投資人不可能單獨投資水泥類股。

七、延伸應用

(一)檢驗指定投資組合所對應的風險與報酬

輸入投資人所選投資組合的各股名稱、各股權重，以及時間區間經過上述矩陣 A、矩陣 B 的運算即可獲得特定投資組合的風險及報酬。

(二)檢驗指定投資組合的效率

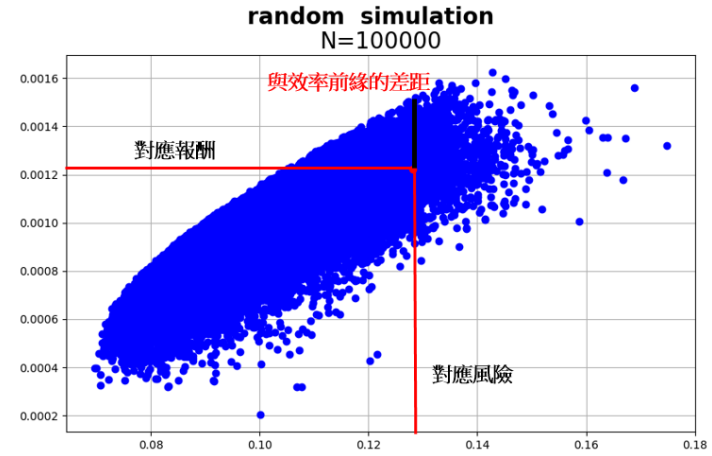
輸入投資人所選投資組合的各股名稱、權重以及時間區間，經過運算後比較該組合與效率前緣(給定風險、報酬最高的亂數點)的垂直距離。

```
import pandas as pd
import yfinance as yf
import numpy as np
import matplotlib.pyplot as plt
from functools import reduce
stocks=['1101.tw', '1102.tw', '1103.tw', '1104.tw', '1108.tw', '1109.tw', '1110.tw']#, '1101B.tw']#股票代碼
specific_weight=[0.1, 0.5, 0.2, 0.2, 0, 0, 0]#指定權重
expected_return_i=[]#平均回報
stocks_return=np.nan#個股回報(日)
for i in range(0, len(stocks)):#擷取資料
  if i==0:
    n=yf.download(stocks[i], start='2019-01-01', end='2020-01-01')
    stocks_return=round(n['Close'].diff()/n['Close'], 7)
    n5=len(n['Close'])
  else:
    n=yf.download(stocks[i], start='2019-01-01', end='2020-01-01')
    na=round(n['Close'].diff()/n['Close'], 7)
    stocks_return=pd.concat([stocks_return, na], axis=1)
stocks_return.columns=stocks#增加各股回報名稱
stocks_return=stocks_return.dropna()#去除第一位空值
for i in range(0, len(stocks)):#平均回報(日)
  na=stocks_return.iloc[:, i].mean()
```

輸入代碼

輸入權重

輸入時間 區間



(三)檢驗特定類股

以亂數點將可行解分布、效率前緣畫出，依據亂數點的相關係數或是比較每年的效率前緣判斷該類股在產業周期位置。