Quantitative Portfolio Management

Assignment #6

Raman Uppal
EDHEC Business School

November 2023

Instructions for each assignment . . . I

- Assignment #1 should be done individually.
- The other assignments are to be done in groups of 4 or 5 students.
  - This means that groups of 1, 2, 3, 6, etc. are not allowed.
  - Diversity in groups is strongly encouraged (people from different countries, different genders, different finance knowledge, and different coding ability, etc.)

Instructions for each assignment . . . II

- Each assignment should be emailed as a Jupyter file
  - To Raman.Uppal@edhec.edu
  - The subject line of the email should be: "QPM: Assignment n," where $n = \{1, 2, \ldots, 8\}$.
  - Assignment n is due before Lecture n, where $n = \{1, 2, \ldots, 8\}$.
  - Assignments submitted late will not be accepted (grade $= 0$), so please do not email me assignments after the deadline.

Instructions for each assignment ... III

- The Jupyter file should include the following (use Markdown):
  - Section "0" with information about your submission:
    - Line 1: QPM: Assignment n
    - Line 2: Group members: listed alphabetically by last name, where the last name is written in CAPITAL letters
    - Line 3: Any comments/challenges about the assignment
  - Section "k" where $k = \{1, 2, \ldots\}$.
    - First type Question k of Assignment n.
    - Then, below the question, provide your answer.
    - Your code should include any packages that need to be imported.

Questions for Assignment 6 . . . I

- In this question, we use the Black-Litterman model to determine the optimal portfolio weights for an investor who is considering investing in AAPL, MSFT, AMZN, NVDA, TESLA, and META.
- Please download prices for these 6 stocks and compute their monthly excess returns starting January 2015 and ending December 2022, assuming that the risk-free rate is 0.
- Use the “Index Weighting” reported in this article from Investopedia to assign the market weights for these assets (you may also be able to get the weights from Yahoo Finance).

Questions for Assignment 6 . . . II

Q6.1 Based on the sample data, compute the Markowitz portfolio weights.

Q6.2 Then, using the market-capitalization weights, obtain the CAPM-implied expected returns.

Q6.3 Then, specify the pick matrix P and the view vector q that captures the following views for each of the assets:

- AAPL: its absolute excess return is expected to be 10% per year.
- MSFT: its absolute excess return is expected to be 5% per year.
- AMZN: no views
- NVDA will outperform TSLA by 2% per year.
- TSLA will underperform META by 1% per year.

Finally, explain your choice for the matrix Ω, which captures the uncertainty about these views.

Questions for Assignment 6 . . . III

Q6.4 Use these views to compute the conditional expected excess return and conditional covariance matrix of excess returns μ_{BL} and Σ_{BL}.

Q6.5 Use μ_{BL} and Σ_{BL} to compute the mean-variance weights and compare them with the weights from the CAPM and the weights based on sample moments.

Discussion of Assignment 6: Initial setup

- We start by loading the libraries we will need.

Code to load required libraries

```
import numpy as np
import pandas as pd
import yfinance as yf
```

Code to save the market-capitalization weights

```
# Store the market-capitalization weights
w_MktCap = pd.Series({"AAPL": 0.071,
          "MSFT": 0.0651,
          "AMZN": 0.0324,
          "NVDA": 0.0284,
          "TSLA": 0.0187,
          "META": 0.0184})
```

Discussion of Assignment 6: Download data

Code to download data

```
# Make a list of the tickers for which we want data
stockdata = ["AAPL", "MSFT", "AMZN", "NVDA", "TSLA", "META"]

# Set the start and end dates
start_date = "2015-01-01"
end_date = "2022-12-31"

# Create empty list to store stock prices (to speed up computation)
dataframes = []

# Download monthly prices from Yahoo finance
for ticker in stockdata:
    Stock_data = yf.download(ticker, start=start_date, end=end_date)
    dataframes.append(Stock_data["Adj Close"])

# Use concatenate to build the dataframe
stock_prices = pd.concat(dataframes, axis=1)
stock_prices.index = pd.to_datetime(stock_prices.index)

# Sample the prices on a monthly frequency
stock_prices = stock_prices.resample('1M').last()
stock_prices.columns = stockdata      # Should get 96 rows x 6 columns
```

Discussion of Assignment 6: Compute return moments

Code to compute returns and its sample moments

```
# Compute the monthly log-returns  (should get 95 rows x 6 columns)
log_ret = np.log(stock_prices / stock_prices.shift(1)).dropna()

# Compute the mean of monthly log returns
mu = pd.Series({"AAPL": np.mean(log_ret["AAPL"]),
          "MSFT": np.mean(log_ret["MSFT"]),
          "AMZN": np.mean(log_ret["AMZN"]),
          "NVDA": np.mean(log_ret["NVDA"]),
          "TSLA": np.mean(log_ret["TSLA"]),
          "META": np.mean(log_ret["META"])})

# Compute the covariance matrix of monthly log returns
Sigma = log_ret.cov()
```

- These moments are reported on the next page.
- All moments are per month, not per year.

Sample moments of returns

Mean of sample returns

	mu_sample
AAPL	0.016789
MSFT	0.020180
AMZN	0.016376
NVDA	0.036379
TSLA	0.023216
META	0.004850

Covariance matrix of sample returns

	AAPL	MSFT	AMZN	NVDA	TSLA	META
AAPL	0.006955	0.003250	0.004066	0.006510	0.007281	0.003174
MSFT	0.003250	0.003824	0.003796	0.005055	0.004351	0.002776
AMZN	0.004066	0.003796	0.008089	0.006791	0.006656	0.004081
NVDA	0.006510	0.005055	0.006791	0.017616	0.007284	0.004440
TSLA	0.007281	0.004351	0.006656	0.007284	0.027660	0.004363
META	0.003174	0.002776	0.004081	0.004440	0.004363	0.009642

Discussion of Assignment: Q6.1

Q6.1 Based on the sample data, compute the Markowitz portfolio weights.

Code to compute Markowitz mean-variance weights

```
# Define a function for Markowitz portfolio weights (risk free = 0)
def w_Markowitz(gamma,sigma,mu):
    inv_sigma = np.linalg.inv(sigma)
    w = (1 / gamma) * inv_sigma @ mu
    return w

# Define the mkt gamma (you could have used some other value)
gamma_mkt = 3.0271189

# Compute Markowitz weights based on sample means and covariances
w_MVU = w_Markowitz(gamma_mkt,Sigma,mu)
```

- The output is printed on the next page

Markowitz mean-variance portfolio weights

- Unscaled (raw) weights

	mu_sample	w_MVU	w_MktCap
AAPL	0.016789	-0.140255	0.071000
MSFT	0.020180	1.889815	0.065100
AMZN	0.016376	-0.319599	0.032400
NVDA	0.036379	0.394155	0.028400
TSLA	0.023216	0.053466	0.018700
META	0.004850	-0.402117	0.018400

Markowitz mean-variance portfolio weights after rescaling

- To allow for a more reasonable comparison, we can rescale the portfolio weights so that they sum to one.

Code to rescale the weights so they sum to one

```
# To compare the wMVU with wMarketCap, we can rescale both sets of
    weights
Result["w_MVU_Rescaled"] = Result["w_MVU"]/np.sum(Result["w_MVU"])
Result["w_Mkt_Rescaled"] = Result["w_MktCap"]/np.sum(Result["w_MktCap"])
Result
```

Markowitz mean-variance portfolio weights after rescaling (continued)

	mu_sample	w_MVU	w_MktCap	w_MVU_Rescaled	w_Mkt_Rescaled
AAPL	0.016789	-0.140255	0.071000	-0.095058	0.303419
MSFT	0.020180	1.889815	0.065100	1.280827	0.278205
AMZN	0.016376	-0.319599	0.032400	-0.216609	0.138462
NVDA	0.036379	0.394155	0.028400	0.267140	0.121368
TSLA	0.023216	0.053466	0.018700	0.036237	0.079915
META	0.004850	-0.402117	0.018400	-0.272536	0.078632

- The table above shows that there are very large differences between
  - the weights of the Markowitz portfolio based on sample moments and
  - the weights based on market capitalization.

Discussion of Assignment: Q6.2

Q6.2 Using the market-capitalization weights, obtain the CAPM-implied expected returns.

Code to compute CAPM-implied expected returns

```
# Step 1 of Black-Litterman: CAPM-implied expected returns

def CAPMimplret(gamma, V,weights):
    exp_ret = gamma * np.dot(V, weights)
    return exp_ret

mu_capm = CAPMimplret(gamma_mkt, Sigma, w_MktCap)

# Save results in a dataframe
Result = pd.DataFrame({
    "mu_sample": mu,
    "mu_capm": mu_capm,
    "w_MVU": w_MVU,
    "w_MktCap": w_MktCap,
    "w_MVU_Rescaled": w_MVU/np.sum(w_MVU),
    "w_Mkt_Rescaled": w_MktCap/np.sum(w_MktCap)
})
Result
```

Output for Q6.2

- The CAPM-implied expected returns are reported in the column titled "mu_capm."
- Comparing this column to the "mu_sample" column, we see that sample mean returns are very different from those from the CAPM.

	mu_sample	mu_capm	w_MVU	w_MktCap	w_MVU_Rescaled	w_Mkt_Rescaled
AAPL	0.016789	0.003683	-0.140255	0.071000	-0.095058	0.303419
MSFT	0.020180	0.002660	1.889815	0.065100	1.280827	0.278205
AMZN	0.016376	0.003603	-0.319599	0.032400	-0.216609	0.138462
NVDA	0.036379	0.005236	0.394155	0.028400	0.267140	0.121368
TSLA	0.023216	0.005510	0.053466	0.018700	0.036237	0.079915
META	0.004850	0.002795	-0.402117	0.018400	-0.272536	0.078632

Discussion of Assignment: Q6.3

Q6.3 Specify the pick matrix P and the view vector q that captures the following views for each of the assets:

- AAPL: its absolute excess return is expected to be 10% per year.
- MSFT: its absolute excess return is expected to be 5% per year.
- AMZN: no views
- NVDA will outperform TSLA by 2% per year.
- TSLA will underperform META by 1% per year.

Finally, explain your choice for the matrix Ω, which captures the uncertainty about these views.

Discussion of Assignment: Q6.3 (continued)

Code for modeling "views" for Black-Litterman model

```
# Define view (q) vector, pick (P) matrix, uncertainty (Omega) matrix

# Important to convert annual views into monthly terms
q = np.array([0.10,0.05,0.02,0.01]) * (1/12)

P = np.array([[1,0,0,0,0,0],
              [0,1,0,0,0,0],
              [0,0,0,1,-1,0],
              [0,0,0,0,-1,1]])

# Build the uncertainty matrix
# Define tau as 1/T
# T = 95 months (8 x 12 monthly prices - 1 to go from prices to returns)
T = log_ret.shape[0]
tau = 1/T

Omega_temp = P @ (tau*Sigma) @ P.T     # first step
diagOmega = np.diag(Omega_temp)        # take the diagonal elements
Omega = np.diag(diagOmega)             # construct a diagonal matrix
```

Output for Q6.3

Code to format and print the matrix Omega

```
np.set_printoptions(formatter={'float': '{: 0.6f}'.format})
print(Omega)
```

Omega matrix

0.000073	0.000000	0.000000	0.000000
0.000000	0.000040	0.000000	0.000000
0.000000	0.000000	0.000323	0.000000
0.000000	0.000000	0.000000	0.000301

- Note that Black and Litterman do not provide a particular approach for constructing the uncertainty matrix, Omega (Ω).
- There are several different methods used in the literature.
- The Omega matrix computed above is the first step for specifying the uncertainty matrix, with many further refinements possible.
  - For example, one may wish to scale the entire matrix; see Idzorek (2007) for a more detailed discussion of this.

Discussion of Assignment: Q6.4

Q6.4 Use these views to compute the conditional expected excess return and conditional covariance matrix of excess returns μ_{BL} and Σ_{BL}.

- First we show how to obtain μ_{BL} (on this slide and the next) and then how to compute Σ_{BL}.

Code to compute the expected return conditional on views

```
# Compute the posterior expected excess returns
# mu_BL is defined in small steps to make it easier to read
term1 = np.linalg.inv(np.linalg.inv(tau*Sigma) +
                      P.T @ np.linalg.inv(Omega) @ P
                      )
term21 = np.linalg.inv(tau*Sigma) @ CAPMimplret(gamma_mkt, Sigma,
    w_MktCap)
term22 = P.T @ np.linalg.inv(Omega) @ q
mu_BL = term1 @ (term21 + term22)
```

Output for Q6.4: μ_{BL}

	mu_ sample	mu_capm	mu_BL	w_MVU	w_MktCap	w_MVU_Rescaled	w_Mkt_Rescaled
AAPL	0.016789	0.003683	0.005938	-0.140255	0.071000	-0.095058	0.303419
MSFT	0.020180	0.002660	0.003907	1.889815	0.065100	1.280827	0.278205
AMZN	0.016376	0.003603	0.005040	-0.319599	0.032400	-0.216609	0.138462
NVDA	0.036379	0.005236	0.007641	0.394155	0.028400	0.267140	0.121368
TSLA	0.023216	0.005510	0.006170	0.053466	0.018700	0.036237	0.079915
META	0.004850	0.002795	0.004533	-0.402117	0.018400	-0.272536	0.078632

- We see that mu_BL is much closer to mu_capm than to mu_ sample;
- This tells us that the Black-Litterman portfolio weights, w_{BL}, will also be closer to the weights based on market capitalizations, w_{MKT}, than to the sample-based w_{MVU}.

Code and output for Q6.4 (continued): Σ_{BL}

Code for the posterior covariance matrix of returns

```
# Posterior return covariance matrix
Sigma_BL = Sigma + np.linalg.inv(np.linalg.inv(tau* Sigma) + P.T @ np.
    linalg.inv(Omega) @ P)
```

Posterior covariance matrix of returns

	AAPL	MSFT	AMZN	NVDA	TSLA	META
AAPL	0.006987	0.003260	0.004080	0.006535	0.007306	0.003188
MSFT	0.003260	0.003842	0.003813	0.005075	0.004367	0.002788
AMZN	0.004080	0.003813	0.008149	0.006826	0.006689	0.004107
NVDA	0.006535	0.005075	0.006826	0.017729	0.007334	0.004466
TSLA	0.007306	0.004367	0.006689	0.007334	0.027787	0.004403
META	0.003188	0.002788	0.004107	0.004466	0.004403	0.009720

Discussion of Assignment: Q6.5

Q6.5 Use μ_{BL} and Σ_{BL} to compute the mean-variance weights and compare them with the weights from the CAPM and the weights based on sample moments.

Code to compute the Black-Litterman portfolio weights

```
# Standard definition for mean-variance portfolio weights,
#   but using mu_BL and sigma_BL, instead of sample moments
w_BL = (1/gamma_mkt) * np.linalg.inv(Sigma_BL) @ mu_BL
```

- The output for all our work is reported in the table below.
  - Note that the weights below have not been scaled to add up to one.

	mu_sample	mu_capm	mu_BL	w_MVU	w_MktCap	w_BL
AAPL	0.016789	0.003683	0.005938	-0.140255	0.071000	0.184012
MSFT	0.020180	0.002660	0.003907	1.889815	0.065100	0.087854
AMZN	0.016376	0.003603	0.005040	-0.319599	0.032400	0.032062
NVDA	0.036379	0.005236	0.007641	0.394155	0.028400	0.030274
TSLA	0.023216	0.005510	0.006170	0.053466	0.018700	-0.011908
META	0.004850	0.002795	0.004533	-0.402117	0.018400	0.046451

Discussion of Assignment: Q6.5 (continued)

- On this slide, we report portfolio weights after they have been rescaled to add up to one.
- Note from the table that
  1. The Black-Litterman weights are very different from the mean-variance weights based on sample moments;
  2. The Black-Litterman weights are similar to the weights based on market capitalizations;
  3. Deviations of the Black-Litterman weights from the capitalization based weights are a result of the views of the investor; e.g., the weight on AAPL is higher because the view about it is optimistic.

	mu_sample	mu_capm	mu_BL	w_MVU_Rescaled	w_Mkt_Rescaled	w_BL_Rescaled
AAPL	0.016789	0.003683	0.005938	-0.095058	0.303419	0.499020
MSFT	0.020180	0.002660	0.003907	1.280827	0.278205	0.238251
AMZN	0.016376	0.003603	0.005040	-0.216609	0.138462	0.086950
NVDA	0.036379	0.005236	0.007641	0.267140	0.121368	0.082100
TSLA	0.023216	0.005510	0.006170	0.036237	0.079915	-0.032293
META	0.004850	0.002795	0.004533	-0.272536	0.078632	0.125971

Bibliography

Idzorek, T. 2007. A step-by-step guide to the Black-Litterman model: incorporating user-specified confidence levels. In *Forecasting expected returns in the financial markets*, 17–38. Elsevier. (Cited on page 20).

End of assignment