Assignment 2: Portfolio Optimisation with Market Data

FMAT3888

Anabel Geraldine
SID: 520360707
Group 6

Unit of Study: FMAT3888 - Projects in Financial Mathematics

University of Sydney
Link for the codes:
https://colab.research.google.com/drive/18Upel5hmkZpPCj6QQrocP7VW7MAGMgjk?usp=sharing
Supervisor: Igor Geninson

1. Executive Summary

This project on portfolio optimization explores strategies for allocating investments across many different asset classes to achieve an ideal balance between risk and return. Using real-world market data, we aim to develop strategies for maximizing returns while minimizing risks under different market conditions. This report explores different asset allocation strategies to determine optimal investment approaches.

We constructed static portfolios aimed at maximizing utility, experimenting with different risk aversion levels and incorporating unhedged International Listed Equity to explore diverse investment options. Additionally, we developed static portfolios along the efficient frontier with a 6% return target and adjusted these to prohibit short selling. Our dynamic portfolio, which yielded the highest returns with the lowest volatility, demonstrated the effectiveness of adaptive strategies in balancing risk and reward. These findings underscore the need for robust parameter estimation and model flexibility, laying a foundation for future strategies in financial portfolio management and supporting the development of sophisticated investment models adaptable to evolving market conditions. These insights might give valuable guidance in constructing resilient portfolios, contributing to more informed and strategic investment decisions.

2. Introduction

The realm of financial investment has increasingly embraced portfolio optimization as a crucial strategy to balance against expected returns, utilizing real-world market data to guide decision-making processes. Portfolio optimization theory is the process of selecting the best combination of assets to maximize returns while minimizing risk according to an investor's goals and risk tolerance *(Vinny and Jeff, 2022)*. This involves using mathematical and statistical models to determine the ideal asset allocation that provides the highest expected return for a given level of risk, or the lowest risk for a given level of expected return. Some optimization theories include Modern Portfolio Theory (MPT), Utility Theory, and the Capital Asset Pricing Model (CAPM) play foundational roles in financial management by outlining methods to maximize investment returns while minimizing risk *(Junaid Qadir., 2014)*. Modern Portfolio Theory (MPT) promotes diversification to create portfolios that maximize returns for a given risk level. Utility Theory focuses on tailoring portfolios to an investor's personal risk preferences, maximizing their satisfaction or "utility" from returns. The Capital Asset Pricing Model (CAPM) links an asset's expected return to its market risk, suggesting that higher risk should lead to higher returns *(Will Kenton, 2014)*. While MPT and CAPM offer market-oriented frameworks, emphasizing diversification and risk-return relationships, Utility Theory adds a personalized layer by considering individual investor preferences.

In this report, we explore these theoretical constructs and apply them practically with real market data. The report will further explore this topic, starting by estimating parameters such as asset means and covariances, segmented into different historical periods to discern how changing economic climates impact asset behaviors and portfolio performance. This project is split into several key phases. This report will further discuss parameter estimation where we determine the necessary statistical inputs from historical financial data, Static Portfolio Optimization where we construct portfolios that maintain a fixed composition over time, using strategies developed from our parameter estimates, Dynamic Portfolio Optimization which allows portfolio adjustments over the investment horizon to adapt to new information or market conditions, and we will explore further complexities by varying risk aversion parameter and the minimum expected return to simulate different financial scenarios and investor profiles. This comprehensive approach will help illustrate the application of theoretical models in real-world scenarios and potentially provide information for future empirical financial studies.

3. Mathematical Setup and Theoretical Result

Parameter Estimation

We first start with ***estimating the parameters*** **a** and ***B*** for monthly returns of eight different assets using market data from 2 time periods: (A) January 2012 to December 2015 and (B) January 2017 to December 2020. These parameters are crucial for modeling asset returns as multivariate normally distributed log-returns, enabling better investment decision-making. The mathematical formulation for mean **a** and covariance ***B*** can be seen below:

$$\mathbf{a} = \frac{1}{T}\sum_{t=1}^{T}\mathbf{X}_t$$

$$\mathbf{B} = \frac{1}{T-1}\sum_{t=1}^{T}(\mathbf{X}_t - \mathbf{a})(\mathbf{X}_t - \mathbf{a})^T$$

Where *Xt* is the vector of log returns for each asset time t, *St* is price of asset at time t, and *T* is total number of observations in each period. This process computes average return and the variability around that average for each asset.

Now we want to ***calculate the n-month return*** for an asset i from month t to t+n. We want to show that the equation has a multivariate normal distribution with mean n**a** and n***B.*** This formulation helps in understanding how returns accumulate over time in a probabilistic manner, under the assumption that returns are log-normally distributed.

Understanding n-month Returns

The n-month return $R^i_{t,n}$ from month t to $t+n$ for asset i is given by:

$$R^i_{t,n} = \frac{S^i_{t+n-1}}{S^i_{t-1}} - 1.$$

Logarithmic Transformation

We transform the n-month returns into logarithmic returns:

$$1 + R_{t,n}^{i} = \frac{S_{t+n-1}^{i}}{S_{t-1}^{i}}$$

Taking logarithms on both sides gives:

$$\log(1 + R_{t,n}^{i}) = \log\left(\frac{S_{t+n-1}^{i}}{S_{t-1}^{i}}\right)$$

This can be rewritten using the properties of logarithms:

$$\log\left(\frac{S_{t+n-1}^{i}}{S_{t-1}^{i}}\right) = \log(S_{t+n-1}^{i}) - \log(S_{t-1}^{i})$$

Sum of Logarithmic Monthly Returns

We know from the definition of log returns:

$$X_t^i = \log\left(\frac{S_t^i}{S_{t-1}^i}\right)$$

Therefore, the difference in logs over n months can be expressed as the sum of n monthly log returns:

$$X_{t,t+n}^{i} = \sum_{k=t}^{t+n-1} \log\left(\frac{S_k^i}{S_{k-1}^i}\right) = \sum_{k=t}^{t+n-1} X_k^i$$

Distribution of the Sum of i.i.d. Variables

Each X_k^i is assumed to be independently and identically distributed (i.i.d.) with a mean a and a covariance matrix B. By the Central Limit Theorem, the sum of these i.i.d. variables, $X_{t,t+n}^{i}$, will also follow a normal distribution:

- Mean: na
- Covariance: nB This is because the sum of means and covariances of i.i.d. variables scales linearly with the number of terms.

Relating to Y^i and Exponential Transformation

Define $Y^i = X_{t,t+n}^{i}$. The distribution of Y^i is $\mathcal{N}(na, nB)$. Then, relating it back to $R_{t,n}^{i}$:

$$\log(1 + R_{t,n}^{i}) = Y^i \Rightarrow R_{t,n}^{i} = e^{Y^i} - 1$$

Moving on, we ***compute the expected returns, covariances, and correlation*** for annual and biennial returns of eight assets, denoted as R(1) and R(2).This calculation will help us understand the joint behavior of asset returns over these periods using the parameters estimated from before. These can be calculated with the formulas below:

$$R_i^{(j)} = e^{Y_i} - 1 \quad \text{with} \quad Y_i \sim \mathcal{N}(12ja_i, 12jb_{ii}) \quad \text{for values of} \quad j = 1, 2.$$

The expected value can be computed by:

$$\mathbb{E}[e^{Y_i} - 1] = \mathbb{E}[e^{Y_i}] - 1 = e^{12ja_i - \frac{1}{2}12jb_{ii}} - 1.$$

The covariance can be computed by:

$$\operatorname{Cov}(e^{Y_i} - 1, e^{Y_l} - 1) = \operatorname{Cov}(e^{Y_i}, e^{Y_l}) = \mathbb{E}[e^{Y_i + Y_l}] - \mathbb{E}[e^{Y_i}]\mathbb{E}[e^{Y_l}].$$

This gives us:

$$\mathbb{E}[e^{Y_i+Y_l}] = e^{12j\left(a_i + a_l - \frac{1}{2}(b_{ii}+b_{ll})\right)}.$$

Thus, the covariance is equal to:

$$\operatorname{Cov}(e^{Y_i} - 1, e^{Y_l} - 1) = e^{12j\left(a_i + a_l - \frac{1}{2}(b_{ii}+b_{ll})\right)} \left(e^{-12jb_{il}} - 1\right).$$

These formulas are derived from the properties of lognormal distribution applied to financial returns, which are crucial for portfolio optimization, allowing investors to understand and mitigate risks while maximizing returns over different time horizons.

Static Portfolio Optimisation

For the ***Static Portfolio Optimization***, we first consider the utility maximization problem for investors allocating their wealth across eight assets over two years. The utility function used is exponential reflecting the investor's risk aversion.

Consider formula:

$$\max \mathbb{E}[U(\mathbf{w}^T\mathbf{R}^{(2)})] = \max \mathbb{E}[-e^{-\gamma \mathbf{w}^T\mathbf{R}^{(2)}}]$$

Subject to the constraint:

$$\sum_{i=1}^{8} w_i = 1,$$

We want to solve it then minimize it:

$$\mathbb{E}[U(R(2)w)] = -\mathbb{E}\left[e^{R(2)w}\right] = -e^{-aR(2)w+\frac{1}{2}w^T BR(2)w}$$
$$-aR(2)w + \frac{1}{2}w^T BR(2)w$$

where:

$\mathbf{w} = (w_1, \dots, w_8)^T$ is the portfolio weight vector, $\mathbf{R}^{(2)}$ represents the two-year returns, γ is the risk aversion parameter set to 1.

Calculating the Efficient Frontier using expected returns, covariances, and the risk aversion parameter. The calculation involves solving for the portfolio weights that maximize the Sharpe ratio, or minimize the risk for a given return.

Now we want to find a portfolio with minimum variance that yields at least 6% return, using a quadratic programming optimization approach. The mathematical formulation is as below:

$$\min \mathbf{w}^T\mathbf{C}\mathbf{w}$$

Here, $\mathbf{C}$ is the covariance matrix of the returns.

Constraints:

Return Constraint:

$$\mathbf{w}^T\boldsymbol{\mu} \geq 0.06$$

\textbf{Sum of Weights Constraint:}

$$\sum_{i=1}^{8} w_i = 1$$

Solution Using Lagrange Multiplier

To solve this, you can apply the method of Lagrange multipliers where you introduce a multiplier for each constraint.

Lagrangian:

$$L(\mathbf{w}, \lambda_1, \lambda_2) = \mathbf{w}^T\mathbf{C}\mathbf{w} - \lambda_1(\mathbf{w}^T\boldsymbol{\mu} - 0.06) - \lambda_2\left(\sum_{i=1}^{8} w_i - 1\right)$$

First Order Conditions: Set the derivative of L with respect to $\mathbf{w}$, λ_1, and λ_2 to zero:

$$\frac{\partial L}{\partial \mathbf{w}} = 2\mathbf{C}\mathbf{w} - \lambda_1\boldsymbol{\mu} - \lambda_2\mathbf{1} = 0$$
$$\frac{\partial L}{\partial \lambda_1} = \mathbf{w}^T\boldsymbol{\mu} - 0.06 = 0$$
$$\frac{\partial L}{\partial \lambda_2} = \sum_{i=1}^{8} w_i - 1 = 0$$

In this report, this problem will be solved using numerical optimizer software to handle complexities of the matrix operation.

Dynamic Portfolio Optimisation

Moving on to **Dynamic Portfolio Optimization**, where an investor adjusts their portfolio weights at the beginning of each year over a two-year investment period. The returns for the assets are given as V1 and V2, and the portfolio weights for the first and second years are represented by w and u, respectively.

The utility maximization problem is set as:

$$\max \mathbb{E}[U(G(w,u))],$$

where the utility function $U(x)$ is defined as $U(x) = -e^{-\gamma x}$ with $\gamma = 1$, and the return of the portfolio over the two years is given by:

$$G(w,u) = (1 + w^T V^1)(1 + u^T V^2) - 1.$$

Constraints: The sum of portfolio weights in each year must equal one:

$$\sum_{i=1}^{8} w_i = 1, \quad \sum_{i=1}^{8} u_i = 1.$$

Formulating the Lagrangian: To handle the constraints, we introduce Lagrange multipliers for each of the constraints. Define the Lagrangian as:

$$L(w,u,\lambda_1,\lambda_2) = \mathbb{E}\left[-e^{-\gamma G(w,u)}\right] - \lambda_1\left(\sum_{i=1}^{8} w_i - 1\right) - \lambda_2\left(\sum_{i=1}^{8} u_i - 1\right).$$

First Order Conditions:

1. Differentiate with respect to (w):

$$\frac{\partial L}{\partial w_i} = \mathbb{E}\left[\frac{\partial}{\partial w_i}\left(-e^{-\gamma G(w,u)}\right)\right] - \lambda_1 = 0.$$

2. Differentiate with respect to (u):

$$\frac{\partial L}{\partial u_i} = \mathbb{E}\left[\frac{\partial}{\partial u_i}\left(-e^{-\gamma G(w,u)}\right)\right] - \lambda_2 = 0.$$

We also want to compute the optimal portfolio weight to minimize variance of a two year portfolio under the condition that the expected return is at least 6% providing a minimum return threshold with the same sum constraints as before.

Lagrangian for the Problem:

Introduce a Lagrange multiplier for the return constraint and formulate the Lagrangian:

$$L(w,u,\lambda) = \mathrm{Var}(G(w,u)) - \lambda\left(\mathbb{E}[G(w,u)] - 0.06\right)$$

Extension: Varying parameter γ and adding ILE_UH

To ***enhance our portfolio optimization strategies***, we are considering two significant adjustments: varying the risk aversion parameter γ and integrating a new asset class, ILE-UH. By adjusting γ, we can explore how different levels of risk tolerance impact the portfolio's performance, allowing for a more tailored investment strategy that matches investor's preference. The addition of ILE_UH aims to diversify the portfolio further and potentially get new market opportunities.

Let's define our adjusted portfolio returns with the inclusion of ILE-UH as (R') and the new asset weights as (w'). The updated objective is to maximize the utility (U) based on the modified risk parameter γ and asset weights:

$$\max U(w', R') = w'^T R' - \frac{\gamma}{2} w'^T \Sigma w'$$

Here, Σ represents the covariance matrix of the returns, now including the variability contributed by ILE-UH. The constraint that the sum of the weights equals one remains:

$$\sum_{i=1}^{n+1} w'_i = 1$$

where n is the number of existing assets plus the new asset ILE-UH. This formulation not only allows us to quantify the effects of varying γ on portfolio risk and return but also to assess how the inclusion of ILE-UH alters the overall risk-return profile.

4. Computational Results

Parameter Estimation

We first ***estimate parameters*** **a** and ***B*** for asset returns using historical market data over two distinct periods as mentioned before. Using python, we approach this by calculating the mean log returns and covariance matrices for each asset class over two intervals.

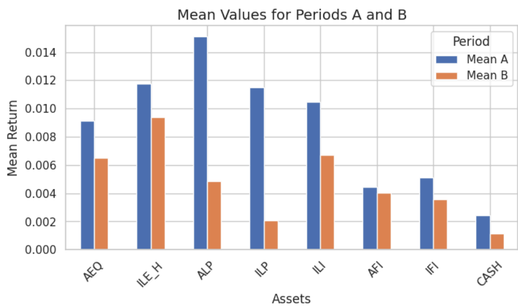

Figure 1. Mean values for Period A & B

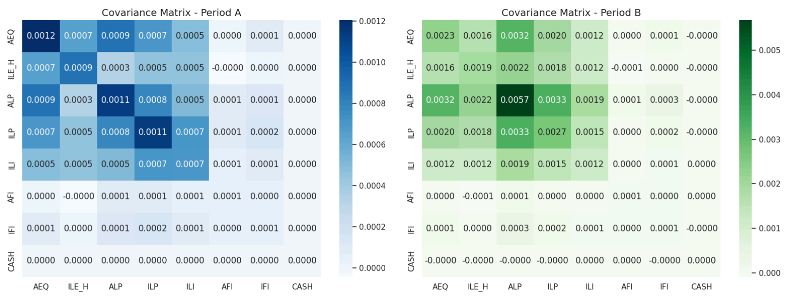

Figure 2. Covariance values for Period A & B

From the result we can see noticeable shifts in mean returns and volatility between the two periods. Assets like AEQ and ALP experienced a decline in mean returns from period A to B, but ILP saw an increase indicating changes in market dynamics. The covariance matrices for both periods show variation in risk profiles.

Next we want to ***express the n-month return*** for each asset in terms of exponential functions using the given log-normal distribution of returns. We assume that the returns compounded over n months can be modeled more accurately, capturing effects of compounding more realistically than a normal distribution might allow. This proving is shown at section 3 of mathematical setup and theoretical result.

We extend the application by modeling the ***joint annual and two-year returns*** for the assets. Using the results from earlier estimations, this is to understand risk and return over more extended periods. We compute this by generating correlation matrix to assess asset co-movements. Then, we compute adjusted covariance matrices for log normal returns, reflecting the risk profile over specific investment horizons. We also prepare a styled table of mean returns, comparing different periods.

Mean Values for Each Data Set

	Mean R1 for A	Mean R2 for A	Mean R1 for B	Mean R2 for B
AEQ	0.107991	0.227643	0.066410	0.137231
ILE_H	0.145933	0.313162	0.106173	0.223620
ALP	0.190769	0.417931	0.024513	0.049627
ILP	0.140273	0.300222	0.008683	0.017442
ILI	0.129300	0.275319	0.076146	0.158091
AFI	0.054294	0.111537	0.049389	0.101217
IFI	0.063146	0.130278	0.043481	0.088852
CASH	0.029625	0.060129	0.013825	0.027840

Figure 3. Mean values for Period A & B (R1 & R2)

Covariance Matrix - R1 for dataset A

	AEQ	ILE_H	ALP	ILP	ILI	AFI	IFI	CASH
AEQ	-0.017602	-0.010000	-0.013727	-0.010437	-0.007357	-0.000228	-0.001034	-0.000039
ILE_H	-0.010000	-0.013643	-0.005369	-0.007157	-0.007228	0.000625	-0.000227	-0.000017
ALP	-0.013727	-0.005369	-0.019077	-0.012690	-0.008096	-0.001367	-0.001806	-0.000062
ILP	-0.010437	-0.007157	-0.012690	-0.017651	-0.010953	-0.001412	-0.002340	-0.000031
ILI	-0.007357	-0.007228	-0.008096	-0.010953	-0.010482	-0.000711	-0.001311	-0.000023
AFI	-0.000228	0.000625	-0.001367	-0.001412	-0.000711	-0.000836	-0.000530	-0.000020
IFI	-0.001034	-0.000227	-0.001806	-0.002340	-0.001311	-0.000530	-0.000748	-0.000015
CASH	-0.000039	-0.000017	-0.000062	-0.000031	-0.000023	-0.000020	-0.000015	-0.000004

Covariance Matrix - R2 for dataset A

	AEQ	ILE_H	ALP	ILP	ILI	AFI	IFI	CASH
AEQ	-0.042908	-0.025293	-0.036032	-0.026264	-0.018356	-0.000533	-0.002435	-0.000088
ILE_H	-0.025293	-0.035645	-0.014625	-0.018652	-0.018654	0.001511	-0.000554	-0.000040
ALP	-0.036032	-0.014625	-0.053735	-0.034300	-0.021708	-0.003430	-0.004570	-0.000153
ILP	-0.026264	-0.018652	-0.034300	-0.045590	-0.028087	-0.003392	-0.005667	-0.000073
ILI	-0.018356	-0.018654	-0.021708	-0.028087	-0.026625	-0.001692	-0.003147	-0.000054
AFI	-0.000533	0.001511	-0.003430	-0.003392	-0.001692	-0.001857	-0.001187	-0.000044
IFI	-0.002435	-0.000554	-0.004570	-0.005667	-0.003147	-0.001187	-0.001690	-0.000033
CASH	-0.000088	-0.000040	-0.000153	-0.000073	-0.000054	-0.000044	-0.000033	-0.000009

Covariance Matrix - R1 for dataset B

	AEQ	ILE_H	ALP	ILP	ILI	AFI	IFI	CASH
AEQ	-0.030805	-0.022899	-0.041150	-0.025135	-0.016321	-0.000353	-0.001599	0.000028
ILE_H	-0.022899	-0.027975	-0.029870	-0.023540	-0.016681	0.001159	-0.000408	0.000051
ALP	-0.041150	-0.029870	-0.069019	-0.040513	-0.024982	-0.001569	-0.003872	0.000043
ILP	-0.025135	-0.023540	-0.040513	-0.032048	-0.019472	-0.000151	-0.002053	0.000011
ILI	-0.016321	-0.016681	-0.024982	-0.019472	-0.016498	-0.000266	-0.001231	-0.000010
AFI	-0.000353	0.001159	-0.001569	-0.000151	-0.000266	-0.000740	-0.000497	-0.000010
IFI	-0.001599	-0.000408	-0.003872	-0.002053	-0.001231	-0.000497	-0.000743	0.000001
CASH	0.000028	0.000051	0.000043	0.000011	-0.000010	-0.000010	0.000001	-0.000004

Covariance Matrix - R2 for dataset B

	AEQ	ILE_H	ALP	ILP	ILI	AFI	IFI	CASH
AEQ	-0.069115	-0.053500	-0.088223	-0.053442	-0.037195	-0.000790	-0.003557	0.000060
ILE_H	-0.053500	-0.067678	-0.066809	-0.051978	-0.039436	0.002692	-0.000941	0.000115
ALP	-0.088223	-0.066809	-0.140125	-0.082092	-0.054461	-0.003371	-0.008264	0.000089
ILP	-0.053442	-0.051978	-0.082092	-0.064186	-0.041895	-0.000319	-0.004318	0.000023
ILI	-0.037195	-0.039436	-0.054461	-0.041895	-0.037940	-0.000600	-0.002764	-0.000022
AFI	-0.000790	0.002692	-0.003371	-0.000319	-0.000600	-0.001629	-0.001089	-0.000021
IFI	-0.003557	-0.000941	-0.008264	-0.004318	-0.002764	-0.001089	-0.001618	0.000003
CASH	0.000060	0.000115	0.000089	0.000023	-0.000022	-0.000021	0.000003	-0.000008

Figure 4. Covariance values for Period A & B (R1 & R2)

The result shows insightful contrast in asset performance. The mean returns in Interval A generally appear higher than those in Interval B. This difference in mean returns indicates shifts in market dynamics and investor sentiment over time. The covariance values in Interval A are generally lower compared to Interval B, reflecting potentially lower volatility or a more stable market environment in the former.

Static Portfolio Optimisation

Now we want to maximize an investor's ***utility from a two year investment.*** The investor is assumed to invest all wealth statically, meaning allocation remains fixed over the investment period. The utility function used is exponential reflecting the investor's risk aversion. Based on the formula derived, we can use python to compute it. This time we also compute the solution for a new period (C) using market data from January 2020 to December 2021. For period a and b we set the boundaries to 0 to 1.

```
Optimal Weights for Period A:
AEQ: 0.0000
ILE_H: 0.0000
ALP: 1.0000
ILP: 0.0000
ILI: 0.0000
AFI: 0.0000
IFI: 0.0000
```

```
Optimal Weights for Period B:
AEQ: 0.0000
ILE_H: 1.0000
ALP: 0.0000
ILP: 0.0000
ILI: 0.0000
AFI: 0.0000
IFI: 0.0000
```

```
Maximum utility Period C:
2.77555756e-17
1.0
2.77555756e-17
0.0
0.0
5.55111512e-17
0.0
```

Figure 5. Optimal Weight and maximum utility for Period A,B, and C

For Period A, it suggests investing entirely in ALP, with no allocation to any other asset classes. As for Period B, the optimal allocation shifts entirely to ILE_H. The stark difference in optimal allocations between Period A and B highlights significant changes in market dynamics or asset performances over time.

Period C displays almost negligible utility but spreads investment across multiple assets, indicating substantial potential for return as shown by the maximum utility return. However, this approach might not be ideal due to the extremely low utility values, which could suggest that the return potential does not compensate for the risk or the portfolio is not optimized to the investor's risk preferences.

Given the risk-averse nature of the utility function , these results suggest a 'winner-takes-all' approach rather than diversification. Ideally, we want to achieve a balance between risk and return while managing market volatility. We should diversify our investment approach, explore varying risk aversion parameters, and consider alternative utility functions.

We compute the efficient frontier for the market consisting of these 8 assets. We can compute it by either using our portfolio and simulated portfolio and both show similar results.

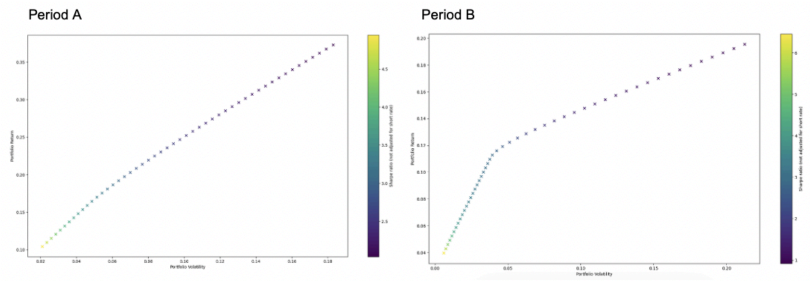

Figure 6. Efficient Frontier for Period A and B

The efficient frontier for Period A shows a smooth increase in return with rising volatility, suggesting balanced diversification. In contrast, Period B's frontier has a kink, indicating fewer diversification options and dominance of certain assets for higher returns at lower risk levels. Now we want to compute and find a ***portfolio with minimum variance*** that yields at least 6% return using the formulate derived from section 3.

Period A - Minimized variance for 6% returns: (SD = 0.002891)

Assets	Optimal Weights
Australian Equities (AEQ)	-0.007
International Listed Equity - Hedged (ILE)	0.006
Australian Listed Property (ALP)	0.004
International Listed Property (ILP)	-0.001
International Listed Infrastructure (ILP)	0
Australian Fixed Income	-0.016
International Fixed Income	0.015
CASH	0.999

Period B - Minimized variance for 6% returns: (SD = 0.008661)

Assets	Optimal Weights
Australian Equities (AEQ)	0.029
International Listed Equity - Hedged (ILE)	0.027
Australian Listed Property (ALP)	-0.02
International Listed Property (ILP)	-0.044
International Listed Infrastructure (ILI)	0.058
Australian Fixed Income (AFI)	0.083
International Fixed Income (IFI)	0.123
CASH	0.744

Figure 7. Minimized Variance for Period A and B

For a minimized variance portfolio targeting a 6% return, Period A heavily relies on Cash with minimal allocation to other assets, achieving a low standard deviation of 0.002891. In Period B, the portfolio diversifies more significantly across multiple asset classes with reduced reliance on Cash, resulting in a higher standard deviation of 0.008661, indicating increased risk. We can also ***adjust for no short selling,*** though this approach may not achieve the absolute minimum variance, reallocating weight from cash ensures the portfolio meets the 6% return target. As a result, this strategy yields returns of 6.19% and 6.84%, respectively.

Dynamic Portfolio Optimisation

We will now examine an investor who allocates their wealth across eight different asset classes over a two-year investment horizon. The investor has the opportunity to ***adjust the***

portfolio weights at the beginning of the second year. The objective is to maximize the expected utility of the two-year portfolio return G(w,u) subject to constraints on the weights.

The approach involves applying a Finite Markov Decision Process to model portfolio optimization as an agent-based interaction with an uncertain environment. Using a Constant-α\alphaα Monte Carlo setup, this method allows iterative learning to optimize portfolio decisions under incomplete information about the environment's behavior. Key parameters include a learning rate $\alpha=0.0002$ to balance new versus historical data, an exploration rate $\epsilon=0.1$ to encourage the agent to try new actions, and M=10,000 iterations to ensure sufficient training. Conducted over the period from January 2016 to December 2017, this process generated 50 portfolio simulations at two points in time.

The results indicate that the optimal dynamic portfolio generated using the Constant-α Monte Carlo method, with a risk-aversion factor of 1, allocates the following weights:

- **Year 1**, [0.52,0.08,0.02,0.06,0.11,0.07,0.11,0.02][0.52, 0.08, 0.02, 0.06, 0.11, 0.07, 0.11, 0.02][0.52,0.08,0.02,0.06,0.11,0.07,0.11,0.02]
- **Year 2**, [0.08,0.19,0.13,0.07,0.01,0.02,0.24,0.24][0.08, 0.19, 0.13, 0.07, 0.01, 0.02, 0.24, 0.24][0.08,0.19,0.13,0.07,0.01,0.02,0.24,0.24],

Resulting in expected returns of 0.10924. However, when compared to the optimal static portfolio over this period, which yielded returns of 0.2236, the static portfolio produced significantly higher returns.The higher returns in the static approach could indicate market stability over this period, where a consistent asset allocation outperformed the dynamically adjusted one.

For the minimum variance with ***expected return is at least 6%,*** the result shows

- Year 1, [0.02,0.03,0.09,0.01,0.09,0.21,0.43,0.13][0.02, 0.03, 0.09, 0.01, 0.09, 0.21, 0.43, 0.13][0.02,0.03,0.09,0.01,0.09,0.21,0.43,0.13]
- Year 2, [0.03,0.11,0.15,0.05,0.21,0.42,0.03,0.01][0.03, 0.11, 0.15, 0.05, 0.21, 0.42, 0.03, 0.01][0.03,0.11,0.15,0.05,0.21,0.42,0.03,0.01]

With expected returns of 0.10187. This dynamic allocation yielded greater returns than the static portfolio optimization, highlighting the advantage of a strategy that adapts to changing market conditions. By rebalancing the portfolio weights each year, the dynamic approach effectively captures evolving asset performance while maintaining a low variance, providing a superior balance of risk and return over the static strategy.

Extension: Varying parameter γ and adding ILE_UH

To further explore this, we adjusted several components: varying parameter γ and adding ILE_UH. In Python, we modify the mean returns and covariance matrix of the assets to include ILE_UH. Then, iterate over a range of γ values to calculate the optimal portfolio weights for each scenario using a loop.

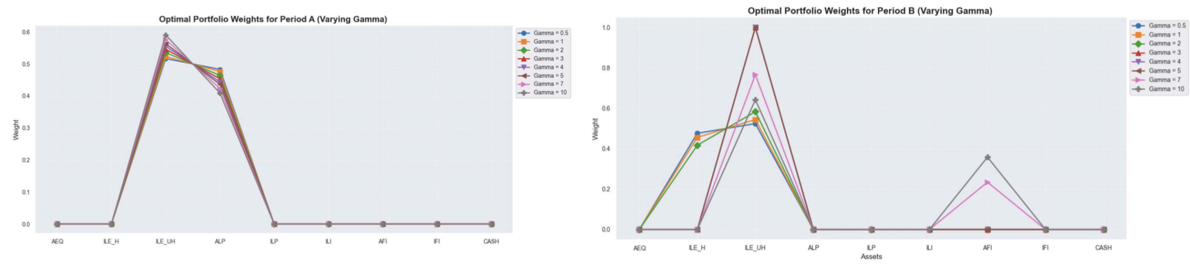

Figure 8. Period A and B after adjustments

The results from varying the parameter γ and adding the asset ILE_UH shows shifts in portfolio allocations for both periods A and B. As γ increases, there is a higher level of risk aversion, the portfolios tend to put more weight toward lower-risk assets. The addition of ILE_UH allows the portfolios to diversify further, enhancing the risk-return profile by providing an asset that can balance returns and risk more effectively across different levels of γ. This approach appears beneficial as it introduces flexibility in asset allocation, particularly under high-risk aversion scenarios, allowing for improved control over portfolio volatility without sacrificing returns.

5. Conclusion

In this research, we incorporate unhedged International Listed Equity to diversify and tap into global market opportunities. We targeted a specific 6% return on the Efficient Frontier, modifying these portfolios to exclude short selling for practical investor applicability. *Our dynamic portfolio, adjusted across two periods,* outperformed these static approaches by achieving higher returns with lower volatility, demonstrating the effectiveness of dynamic strategies in adapting to market changes and optimizing portfolio performance in diverse economic conditions.

In our analysis of portfolio optimization, both static and dynamic strategies were explored using varied computational methods. Static optimization generally yielded higher returns, demonstrating its effectiveness in stable market conditions where set allocations maintain value. Dynamic optimization showed better adaptability to market fluctuations by adjusting allocations, which could potentially outperform static models under volatile market conditions.

Suggestions for future studies include further exploration into the integration of real-time data to enhance dynamic optimization models and deeper investigation into alternative risk measures beyond variance. The extension involving varying risk aversion parameters and the addition of new asset classes, like ILE_UH, revealed potential for tailored investment strategies that balance risk and return more effectively. Limitations such as model dependency on historical data and assumptions in risk modeling must be addressed to improve the robustness and applicability of portfolio optimization models in real-world scenarios.

6. References

Junaid Qadir. (2014). *Optimization theory*. Retrieved from https://www.sciencedirect.com/topics/computer-science/optimization-theory#:~:text=In%20constrained%20nonlinear%20optimization%2C%20when,%E2%80%93Tucker%20(KKT)%20conditions.

Vinny and Jeff, (2022). The benefits of portfolio optimization: Why you should consider this investment strategy. Retrieved from https://www.globalbankingandfinance.com/the-benefits-of-portfolio-optimization-why-you-should-consider-this-investment-strategy/#:~:text=Ultimately%2C%20portfolio%20optimization%20allows%20you,according%20to%20their%20life%20circumstances.

Will Kenton, (2014). *Capital Asset Pricing Model (CAPM)*. Retrieved from https://www.investopedia.com/terms/c/capm.asp

7. Appendix

Appendix 1. Code for Static Optimization

```
initial = np.ones(8)/8
risk_aversion = 1
####################
#### INTERVAL A ####
####################

maxutil_A = maximise_utility(initial, two_returnsA, risk_aversion) #Weights of the maximum utility portfolio
maxutil_returnsA = port_returns(two_returnsA, maxutil_A) #Log returns from the maximum utility portfolio

####################
#### INTERVAL B ####
####################

maxutil_B = maximise_utility(initial, two_returnsB, risk_aversion) #Weights of the maximum utility portfolio
maxutil_returnsB = port_returns(two_returnsB, maxutil_B) #Log returns from the maximum utility portfolio


print("Values for both periods A and B:")
maxutil_A, maxutil_returnsA, maxutil_B, maxutil_returnsB
```

Appendix 2. Code for Dynamic Optimization

```
    #INITIALISING FOR ALL STATES AND ACTIONS
    q_s0_a0 = np.zeros(num_actions)
    q_s1_a1 = np.zeros((num_states, num_actions))
    returns_s0_a0 = np.zeros((num_actions))
    returns_s1_a1 = np.zeros((num_states, num_actions))

    #initialise abrbitrary policies for each state
    policy0 = np.ones(num_actions)/num_actions
    policy1 = np.ones((num_states, num_actions))
    policy1 /= num_actions

    def get_util_rewards(initial_weight, rebalanced_weight, expected_returns):
        wealth = (1 + np.sum(initial_weight * expected_returns)) * (1 + np.sum(rebalanced_weight * expected_returns)) - 1
        reward = np.exp(-1 * risk_aversion * wealth)
        return -1 * reward

    for m in range(M):
        episode = [0]
        action = 0

        for time_step in range(2):
            #generate an episode based on policy
            if time_step == 0:
                action = np.random.choice(num_actions, p = policy0)
            else:
                action = np.random.choice(num_actions, p = policy1[action])
            episode.append(action)
```

```
        G = 0
        for step in range(len(episode) - 1):
            G = risk_aversion * G + get_util_rewards(initial_weights[episode[step]], rebalanced_weights[episode[step+1]], expected_returns)
            if step == 0:
                q_s0_a0[episode[step+1]] += alpha * (G - q_s0_a0[episode[step+1]])
            else:
                q_s1_a1[episode[step], episode[step+1]] += alpha * (G - q_s1_a1[episode[step], episode[step+1]])

            if step == 0:
                A_asterix = np.argmax(q_s0_a0)

                for i in range(len(policy0)):
                    if i == A_asterix:
                        policy0[i] = 1 - epsilon + (epsilon/len(policy0))
                    else:
                        policy0[i] = epsilon/len(policy0)
            else:
                A_asterix = np.argmax(q_s1_a1[episode[step]])

                for i in range(len(policy1[episode[step]])):
                    if i == A_asterix:
                        policy1[episode[step], i] = 1 - epsilon + (epsilon/len(policy1))
                    else:
                        policy1[episode[step], i] = epsilon/len(policy1)
```

Appendix 3. Code for Extension: varying parameter gamma and adding ILE_UH

```
# Optimization function
def optimize_portfolio(mean_returns, cov_matrix, gamma):
    num_assets = len(mean_returns)

    def objective(weights):
        portfolio_return = np.dot(weights, mean_returns)
        portfolio_variance = np.dot(weights.T, np.dot(cov_matrix, weights))
        certainty_equivalent = portfolio_return - (gamma / 2) * portfolio_variance
        return -certainty_equivalent

    constraints = {'type': 'eq', 'fun': lambda weights: np.sum(weights) - 1}
    init_guess = np.repeat(1 / num_assets, num_assets)
    bounds = [(0, 1) for _ in range(num_assets)]

    result = minimize(objective, init_guess, method='SLSQP', constraints=constraints, bounds=bounds)

    return result.x

# Varying gamma values
gamma_values = [0.5, 1, 2, 3, 4, 5,7, 10]
weights_A = []
weights_B = []

for gamma in gamma_values:
    weights_A.append(optimize_portfolio(mean_A, covariance_A, gamma))
    weights_B.append(optimize_portfolio(mean_B, covariance_B, gamma))
```