GROUP WORK PROJECT # 3 MScFE 652: PORTFOLIO MANAGEMENT
Group Number: 8244

FULL LEGAL NAME	LOCATION (COUNTRY)	EMAIL ADDRESS	MARK X FOR ANY NON-CONTRIBUTING MEMBER
Shivansh Kumar	India	shivansh.business23@gmail.com	
Danish Rizwan	Pakistan	danishrizwan.dr@gmail.com	
Bharat Swami	India	bharatswami1299@gmail.com	

Statement of integrity: By typing the names of all group members in the text boxes below, you confirm that the assignment submitted is original work produced by the group (excluding any non-contributing members identified with an "X" above).

Team member 1	**Shivansh Kumar**
Team member 2	**Danish Rizwan**
Team member 3	**Bharat Swami**

Use the box below to explain any attempts to reach out to a non-contributing member. Type (N/A) if all members contributed.
Note: You may be required to provide proof of your outreach to non-contributing members upon request.

(N/A)

Step 1

Improvement with denoising

Portfolio optimization aims to balance risk and return through optimal asset allocation. Traditional methods, such as the Markowitz mean–variance framework, rely heavily on accurate estimates of expected returns and covariances. However, noise often ruins financial data due to market microstructure effects, statistical estimation errors, and exogenous shocks. Denoising techniques can extract the underlying signal, leading to more robust portfolio construction. In our report, we examined three advanced techniques for covariance estimation and denoising: Ledoit-Wolf (LW) shrinkage estimator, Oracle Approximating Shrinkage (OAS) estimator, and de Prado's Constant Residual Eigenvalue Method (CREM) (de Prado).

Ledoit-Wolf (LW) Shrinkage Estimator:

The LW shrinkage estimator improves the sample covariance matrix by shrinking it toward a structured target matrix. This technique addresses the high variability of the empirical estimator, especially in high-dimensional settings ("LedoitWolf — scikit-learn 1.6.1 documentation").

The LW estimator is given by:

$$\widehat{\Sigma}_{LW} = \lambda I + (1 - \lambda)\widehat{\Sigma}$$

Where:

- $\widehat{\Sigma}$ is the empirical covariance matrix
- I is a shrinkage target (commonly the identity matrix scaled by the average eigenvalue or a constant correlation matrix)
- $\lambda \in [0, 1]$ is the Shrinkage intensity, determined by minimizing the mean squared error between the true covariance and the estimator.

The optimal shrinkage intensity is estimated analytically from the data, which balances the trade-off between bias and variance.

Oracle Approximating Shrinkage (OAS) Estimator:

The OAS estimator is an improvement over the LW estimator, particularly for small sample sizes. It provides an analytically derived optimal shrinkage parameter that further minimizes estimation error.

Just like LW estimator, the OAS estimator shrinks the empirical covariance matrix toward a target I ("OAS — scikit-learn 1.6.1 documentation"):

$$\hat{\Sigma}_{OAS} = \lambda_{OAS} I + (1 - \lambda_{OAS})\hat{\Sigma},$$

With the optimal Shrinkage intensity λ_{OAS} determined by:

$$\lambda_{OAS} = \frac{(1-2/p)tr(\hat{\Sigma}^2) + tr(\hat{\Sigma})^2}{(n+1-2/p)[tr(\hat{\Sigma}^2) - \frac{tr(\hat{\Sigma})^2}{p}]}$$

- p is the number of assets
- n is the number of observations
- tr(.) denotes the trace operator

The OAS estimator has been shown to yield improved performance in out-of-sample tests, especially when the sample size is small relative to the number of assets.

De Prado's Constant Residual Eigenvalue Method:

CREM enhances the estimation of covariance matrices by filtering out noise using eigenvalue analysis. The method is based on the observation that noisy data manifests predominantly in the smaller eigenvalues of the covariance matrix (de Prado).

Let's understand the Steps of CREM:

1> With a given covariance matrix $\hat{\Sigma}$, we will perform the eigenvalue decomposition:

$$\hat{\Sigma} = V\Lambda V^T$$

Where:

- V is the matrix of eigenvector
- $\Lambda = diag(\lambda_1, \lambda_2, \lambda_3 \ldots\ldots, \lambda_n)$contains the eigenvalues sorted in descending order.

2> After that, we perform Constant Residual Eigenvalue in which, below a certain threshold, the eigenvalues are considered to represent noise. CREM establishes a constant residual eigenvalue as the noise floor.

3> Next step is to perform eigenvalue Adjustment:

$$\bar{\lambda} = \lambda_i, \, if\, \lambda_i > \lambda_c \, OR \, \lambda_c, \, if\, \lambda_i \le \lambda_c$$

4> At last we will reconstruct the denoised covariance matrix :

$$\bar{\Sigma} = V\bar{\Lambda}V^T, \text{ with } \bar{\Lambda} = diag(\bar{\lambda}_1, \bar{\lambda}_2, \ldots\ldots, \bar{\lambda}_n)$$

Improvement Through Clustering

Introduction

Portfolio optimization is a fundamental problem in modern finance. The challenge is effectively allocating an investor's funds across diversified assets to maximize returns while minimizing risk. One innovative approach to improving portfolio optimization is clustering, i.e, grouping similar data points based on their characteristics. By incorporating clustering into portfolio optimization, we can make more informed investment decisions, leading to improved risk-adjusted returns.

Clustering in Portfolio Optimization

Clustering refers to the process of grouping similar data points. This involves clustering assets based on their historical performance and correlation. One popular method is Hierarchical Risk Parity (HRP), which uses correlation coefficients to measure the similarity between asset time series. HRP utilizes an agglomerative nesting approach: Each asset starts as an individual cluster, and assets with higher correlation gradually merge into larger clusters until similar assets are grouped together.

Several other clustering methods can be employed for portfolio optimization, including:

- **K-Means Clustering:** A widely used unsupervised learning algorithm that groups assets into k clusters based on the closest centroid. The algorithm iteratively adjusts centroids to minimize the sum of squared distances within each cluster.
- **Weighted K-Means Clustering:** A variation of K-Means that assigns different weights to data points, allowing for more refined clustering outcomes, particularly when some assets carry more significance than others.
- **Other Clustering Methods:** Various other clustering methods exist, such as Gaussian mixture models, Modha-Spangler weighting, and Gower's distance, though each comes with its limitations, such as strong parametric assumptions or arbitrary weighting criteria. (Guba et al.)

Benefits of Clustering in Portfolio Optimization

1. **Enhanced Diversification:** By clustering assets based on their correlation structure, investors can allocate funds more effectively, ensuring exposure to a well-diversified set of assets and reducing concentration risk. (Ding, Yuan, et al.)

2. **Risk Reduction:** Clustering allows for the identification of asset groups that exhibit similar behaviors, leading to a better understanding of risk factors and improved risk mitigation strategies.
3. **Improved Computational Efficiency:** Traditional mean-variance optimization can be computationally expensive for large portfolios. Clustering reduces the complexity by first grouping similar assets, thereby simplifying the optimization process.
4. **Robust Portfolio Construction:** By leveraging hierarchical clustering techniques like HRP, portfolios can be constructed in a way that is less sensitive to estimation errors compared to traditional Markowitz optimization.
5. **Better Adaptability to Market Changes:** Clustering enables dynamic portfolio adjustments by re-evaluating asset groupings as market conditions evolve, leading to more adaptive and resilient portfolios.

Integration with Machine Learning and Optimization Algorithms

Clustering techniques can be further enhanced by integrating machine learning and optimization algorithms such as:

- **Neural Networks and Genetic Algorithms:** These methods provide alternative approaches to optimization by leveraging pattern recognition and evolutionary strategies to refine portfolio selection.
- **Classification Techniques:** These supervised learning methods can help in categorizing assets into predefined risk-return categories, assisting investors in selecting suitable investment options.

Conclusion

The use of clustering techniques in financial portfolio optimization presents significant advantages in terms of diversification, risk reduction, and computational efficiency. By leveraging methods such as HRP, K-Means, and machine learning-based classification, investors can enhance their decision-making processes and construct portfolios that are more robust and adaptive to market changes. As financial markets continue to evolve, clustering-based approaches will likely play an increasingly vital role in optimizing investment strategies and improving overall portfolio performance.

Improvement through Backtesting

Introduction

Backtesting is the process of analysing a trading or investment strategy's performance using previous data prior to implementing it in actual markets. Assures that a strategy is sound, lucrative, and risk-conscious prior to implementation. In the absence of backtesting, traders and portfolio managers run the danger of employing tactics that don't work in actual market circumstances.

Backtesting Features

- Assessment of Performance: Calculates the Sharpe ratio, maximum drawdown, returns, volatility, etc. Aids in determining whether a strategy performs better than an alternate approach or a benchmark index.
- Stress testing and risk management: Checks the robustness of a strategy by simulating worst-case scenarios, such as the financial crisis of 2008. Evaluates tail risks, drawdowns, and expected shortfall (CVaR).
- Optimisation of Parameters: Adjusts hyperparameters such as asset weightings, stop-loss thresholds, and rebalancing frequency. Tests on several timescales to prevent overfitting.
- Testing Walk-Forward: Separates data into test and train periods in order to examine stability over time. Make sure the model doesn't rely solely on outdated patterns.

Advantages of Retesting

- Prevents Data Snooping Bias and Overfitting: Guarantees that the model functions in a variety of market circumstances. Avoids depending on arbitrary past patterns.
- Evaluates Strategy Performance Prior to Implementation: Offers a data-driven method for confirming investing choices. Aids in strategy refinement prior to actual capital commitment.
- Enhances Portfolio Management: **Works well with machine learning methods such as clustering and denoising.** Helps improve risk-adjusted returns by fusing traditional finance models with machine learning insights.

Backtesting in Portfolio Optimizations

With its ability to reveal information on risk exposure, asset allocation, and portfolio rebalancing tactics, backtesting is a potent tool for risk management and portfolio optimisation. To ensure real-world applicability and prevent biases, it must be carefully constructed. Portfolio managers can create more robust and effective portfolios by combining traditional backtesting with machine learning methods like denoising and clustering.

Some of the portfolio optimizations can be done as follows-

- Asset Allocation Testing
- Evaluating Portfolio Rebalancing Strategies
- Optimizing Portfolio Weights

Although backtesting is a very powerful tool, we need to be careful what we are using as testing and training data. Like the two most backtesting techniques, which are k-fold Cross-Validation and Walkthrough, have a problem of leakage. And to avoid the problem of leakage, we need to use techniques like purging and Embargo, which basically remove the data points from the training dataset that are overlapping.

Some more challenges associated with backtesting in general-

- Lookahead Bias: Sometimes we use the data for training, which has not occurred yet (i.e., future information) , we do this k-fold cross validation.
- Survivorship Bias: We ignore the asset or group of assets which performed poorly in the past.
- Since backtesting is done on historical data, it is very unlikely the past will repeat itself, which makes backtesting not so robust, but still it provides a good starting point with the best strategy from a bunch of strategies.

<u>Backtesting with Machine Learning for portfolio optimization</u>

By making adaptive risk management and portfolio optimisation techniques possible, machine learning improves backtesting. ML-driven backtesting produces trading models that are more robust, dynamic, and lucrative by utilising methods like deep learning, reinforcement learning, and clustering. To guarantee real-world applicability, however, rigorous model validation and out-of-sample testing are necessary.

Below are a few cases where we can use ML with backtesting-

- Dynamic Portfolio Rebalancing: ML-based backtesting allows adaptive rebalancing depending on shifting market conditions rather than set rebalancing rules:
    - Reinforcement Learning (RL): In reaction to market conditions, algorithms such as PPO and Deep Q-Networks (DQN) dynamically modify asset weights.

  - Clustering (K-Means, DBSCAN): Assists in dynamically adjusting positions and classifying assets into risk-based groups.

  For instance, an RL-based strategy that has been trained on past returns, volatility, and liquidity may beat static rule-based methods in terms of dynamic fund reallocation.

- Feature Engineering & Regime Shifts: By identifying market regimes and modifying methods appropriately, machine learning improves backtesting:
  - Markets can be identified as bull, bear, or sideways using Hidden Markov Models (HMMs).
  - Find macroeconomic changes and structural breaks with Random Forests and XGBoost.

  For instance, a portfolio strategy may switch from mean-reversion tactics in bad markets to momentum-based trading in bull markets.

Conclusion

In portfolio management, backtesting is a crucial tool that enables sound, data-driven decision-making. It greatly lowers uncertainty and improves risk-adjusted performance, despite its inability to make flawless predictions about the future. Better portfolio optimisation can result from using denoising and clustering in conjunction with proper backtesting.

Step 2

For this step, we will work on all the denoising methods that we have discussed in step 1, we will first pick the best portfolio from GWP2, as we have used MVO in GWP2 for portfolio construction, so we think the best performing portfolio has superior risk and reward so it will act as a benchmark for us when we we apply all the improvements wether it is one, two or combination of improvements.

The best weights for use in GWP were:

Table 1: Best weights from GWP2 MVO portfolio

Asset	**Weight**
AAPL	-2.8957
NVDA	-0.4408
TSLA	3.1467
XOM	2.3461
REGN	-2.4048
LLY	-0.2035
JPM	1.452

In our analysis we have evaluated several advanced techniques for refining portfolio construction. The methods include:

- **Shrinkage Estimation (Ledoit-wolf and OAS)**
- **Denoising using constant Residual Eigenvalue Method (CREM)**
- **Clustering**

Now let's discuss the results, we apply multiple improvements individually, as well as combinations of improvements, in Shrinkage estimation, CREM, and backtesting :

- Ledoit-wolf weights: Weights were shifted from the extreme value of the best portfolio weights to a single dominant position shown in table below, In terms of metrics LW estimation resulted in below table.

Table 2: Weights for asset using Ledoit-wolf method

Asset	**Weight**
AAPL	-0.0
NVDA	-0.0
TSLA	0.0247
XOM	0.9753
REGN	0.0
LLY	0.0
JPM	0.0

Table 3: Different matric results for Ledoit-wolf

Metric	**Result**
Returns	66.53%
Variance	53.83%
Sharpe	1.15
Max Drawdown	–60.36%

- OAS weights: This improvement has also somewhat similar weights adjustments **[-0. -0. 0.0272 0.9728 -0. -0. -0.]** and there were some marginal differences in the metrics where **Return = 66.53%, Variance = 53.80%, Sharpe = 1.15, Max Drawdown = –60.23%**

Table 4: Weights for asset using OAS method

Asset	**Weight**

AAPL	-0.0
NVDA	-0.0
TSLA	0.0272
XOM	0.9753
REGN	0
LLY	0.0
JPM	0.0

Table 5: Different matric results for OAS method

Metric	**Result**
Returns	66.53%
Variance	53.80%
Sharpe	1.15
Max Drawdown	–60.23%

- CREM weights: The CREM weights were much cleaner compared to these shrinkage estimators **[0. 0. 0.0321 0.9679 0. 0. 0.]** and the metrics that we have observed were **Return = 65.26%, Variance = 28.87%, Sharpe = 1.13, Max Drawdown = –59.97%.**

Table 6: Weights for asset using CREM method

Asset	**Weight**
AAPL	-0.0
NVDA	-0.0
TSLA	0.0321

XOM	0.9679
REGN	0
LLY	0.0
JPM	0.0

Table 7: Different matric results for CREM method

Metric	**Result**
Returns	65.226%
Variance	28.87%
Sharpe	1.13
Max Drawdown	–59.97%

- Walk-Forward backtesting weights: since in WF backtesting we are going to calculate the weights in each period, we have to average the weights over all the periods, and
    - the average weights are [0.0483 0.0568 0.0667 0.392 0.0367 0.2186 0.1809],
    - Average sharpe_ratio = -0.69924,
    - Average max_drawdown = -0.1172
    - Average expected_shortfall = -0.0427
    - Overall Annualized Return: 40.47%
    - Overall Annualized Volatility: 10.39%
    - Overall Sharpe Ratio: 3.46
    - Overall Maximum Drawdown: -35.78%

Table 8: Weights for asset using Walk-Forward Backtesting method

Asset	**Weight**
AAPL	0.0483
NVDA	0.0568
TSLA	0.0667

XOM	0.392
REGN	0.0367
LLY	0.2186
JPM	0.1809

Table 9: Different matric results for Walk-Forward Backtesting method

Metric	**Result**
Returns	40.47%
Variance	10.39%
Sharpe	3.46
Max Drawdown	-35.78%

- K-Fold Cross Validation Backtesting: Again we have to calculate the average weights since k-fold run for multiple interactions, so the results are-
    - Average sharpe_ratio -0.89732
    - Average max_drawdown -0.21466
    - Average expected_shortfall -0.04236
    - Average Weights from K-Fold Backtesting: [0.   0.   0.3648 0.5333 0.   0.   0.1019]
    - Annualized Return: 59.50%
    - Annualized Volatility: 17.05%
    - Sharpe Ratio: 3.22
    - Maximum Drawdown: -32.77%

Table 10: Weights for asset using K-Fold Backtesting method

Asset	**Weight**
AAPL	0.
NVDA	0.

TSLA	0.3648
XOM	0.5333
REGN	0.
LLY	0.
JPM	0.1019

Table 11: Different matric results for K-Fold Backtesting method

Metric	**Result**
Returns	59.50%
Variance	17.05%
Sharpe	3.22
Max Drawdown	-32.77%

- CPCV (Combined Puring cross validation) Backtesting: similarly as above we calculate the results on average over different iterations-
    - Average sharpe_ratio -0.89732
    - Average max_drawdown -0.21466
    - Average expected_shortfall -0.04236
    - Average Weights from CPCV Backtesting: [0.   0.   0.3648 0.5333 0.   0.   0.1019]
    - Annualized Return: 59.50%
    - Annualized Volatility: 17.05%
    - Sharpe Ratio: 3.22
    - Maximum Drawdown: -32.77%

Table 12: Weights for asset using CPCV method

Asset	**Weight**
AAPL	0.

NVDA	0.
TSLA	0.3648
XOM	0.5333
REGN	0.
LLY	0.
JPM	0.1019

Table 13: Different matric results for CPCV Backtesting method

Metric	**Result**
Returns	59.50%
Variance	17.05%
Sharpe	3.22
Max Drawdown	-32.77%

Note: We have define the Average results and overall results for the backtesting part of the improvement process, the reason of doing such is because in backtesting we silts the dataset into number of folds which makes number of iteration more than one for single dataset by doing training and testing. So when we average over the metrics which have no more significance just showing how on average backtesting is doing whereas overall results show the true results which are based on the average weights for each backtesting method.

Now let's highlight the improvements, whether our improvements are viable or not:

- Single Improvement
  - Shrinkage alone (LW/OAS) already reduces volatility and improves drawdown compared to the original portfolio.
  - Denoising alone (CREM) cuts variance further, offering a substantial reduction in risk.

- ○ Backtesting is not providing the substantial results here. They are reducing the overall results which we can interpret as that the dataset is not much sable over the period. In other worlds we have regimes in the dataset which makes backtesting difficult over the different period.

- Combined Improvements
  - When denoising is combined with clustering, the final weights become more diversified and structured.
  - Final Weights (Denoised + Clustered): **[0.12442581 0.19867419 0.19918288 0. 0.2315 0. 0.24621712]**

Table 14: Weights for asset using (Denoised + Clustered) method

Asset	**Weight**
AAPL	0.12442581
NVDA	0.19867419
TSLA	0.19918288
XOM	0.
REGN	0.2315
LLY	0.
JPM	0.24621712

  - The combined portfolio shows further improvements in key metrics:
    - Sharpe Ratio: Increased from 1.11 (denoised) and 1.17 (clustered) to 1.24.
    - Volatility: Reduced dramatically to 15.66% (from 57.02% and 16.60%).
    - Maximum Drawdown: Improved to –15.17% compared to –59.97% (denoised) and –86.46% (original).
  - We proposed the sequence is to first apply covariance denoising (using CREM and/or shrinkage estimators) to clean the input data. This is followed by clustering to exploit asset similarity and reduce dimensionality. Such a sequence harnesses the strengths of both statistical noise reduction and structural simplification.

Step 3

In this section, we will discuss Multiple Improvements vs single improvements for comparison, we are comparing 4 basic metrics:

- **Sharpe Ratio**: Higher Sharpe indicates a better risk-adjusted return.
- **Volatility**: Lower volatility implies a more stable portfolio.
- **Maximum Drawdown**: Lower drawdown indicates better downside protection.
- **Portfolio**: While we desire high returns, they must be balanced with risk metrics.

Our observation from the outputs of the different improvements is as follows:

- Original Portfolio:
  - Sharpe: 1.60, Volatility: 150.34%, Max Drawdown: –86.46%, Return: 242.83%
  - We have high returns but extremely high volatility and drawdown.
- Single Improvement (LW and OAS):
  - Sharpe: 1.15, Volatility: 53.80%, Max Drawdown: –60.23%, Return: 66.53%
  - Using LW and OAS resulted in a Significant reduction in variance and drawdown, though return and risk–reward metrics drop.
- Single Improvement (CREM):
  - Sharpe: 1.13, Volatility: 28.87%, Max Drawdown: -59.97%, Return: 65.26%
  - Using CREM we have a drop in Sharpe, volatility, and max drawdown but there was no significant change in returns.
- Single Improvement (Clustered only):
  - Sharpe: 1.17, Volatility: 16.60%, Max Drawdown: –15.17%, Return: 21.41%
  - Clustering drastically lowers volatility and drawdown; however, returns are modest.
- Combined Improvements (Denoised + Clustered):
  - Sharpe: 1.24, Volatility: 15.66%, Max Drawdown: –15.17%, Return: 21.41%
  - The combination outperforms individual applications by maintaining a low risk profile while improving the risk–reward balance relative to clustering alone. The slight increase in the Sharpe ratio (1.24 vs. 1.17) suggests that combining the techniques helps harness the advantages of both denoising and clustering.

- Backtesting improvements:
  - As we have noted that we are not able to improve our metrics using the backtesting.
  - And this is because of the presence of regimes in the dataset, also the presence of regime makes the dataset non-stationary, so volatility changes over time.

After going through multiple single as well as combinational improvements using denoising, we have concluded that single improvements (using either shrinkage/denoising or clustering) offer benefits in reducing risk measures such as volatility and maximum drawdown, combining these improvements yields a more robust portfolio.

In combination, we specifically mention the combined denoised and clustered portfolio:

- Offers a lower volatility (15.66% vs. 57.02% in the denoised-only case).
- Achieves the best maximum drawdown (–15.17% compared to –59.97% originally).
- Improves the Sharpe ratio to 1.24, indicating enhanced risk-adjusted performance.

Thus, in our experience, the optimal sequence should be first apply covariance denoising and then follow with asset clustering to exploit correlation and further reduce dimensionality and solidify the portfolio structure.

Step 4

We did performance evaluation of model improvements. The model was trained on data spanning from January 1, 2022, to December 31, 2024. The model was then tested on two weeks of daily data between the GWP2 date and the GWP3 date.

The following cases were trained for and tested:

- **Original Portfolio**: Baseline model without enhancements.
- **Ledoit-Wolf**: Shrinkage estimator for covariance matrix.
- **OAS Portfolio**: Oracle Approximating Shrinkage method.
- **Out of Sample**: Performance assessment on unseen data.
- **Improved by Denoising**: Noise reduction techniques applied to the covariance matrix.
- **Improved by Clustering**: Asset clustering strategies used for portfolio diversification.
- **Improved by Clustering + Denoising**: Combination of clustering and denoising to enhance portfolio stability.
- **Improvement by Backtesting:** We have used three different types of backtesting methods to improve the portfolio weights to produce great returns. And three methods are-
    - Walk-Forward Backtesting
    - K-Fold Cross Validation
    - Combined Purged Cross Validation.

The effectiveness of each method was determined by evaluating key performance indicators such as:

- **Return on Investment (ROI)**
- **Risk (Standard Deviation/Volatility)**
- **Sharpe Ratio**
- **Maximum Drawdown**

Results

- The **Original Portfolio** served as a baseline and exhibited moderate returns with high volatility.
- **Ledoit-Wolf** and **OAS Portfolio** showed improved stability but had slightly reduced returns.
- **Denoising** techniques enhanced portfolio stability, reducing variance and improving the Sharpe Ratio.
- **Clustering** provided better diversification, reducing drawdowns while maintaining returns.
- The **combination of Clustering and Denoising** yielded the best results, optimizing risk-adjusted returns and demonstrating the most consistent performance in out-of-sample testing.

Table 15: showing all the results combined

Combination	Portfolio Return	Portfolio Volatility	Sharpe Ratio	Max Drawdown
Original Portfolio	272.4%	150.3%	1.78	-86.4%
Ledoit-Wolf	66.5%	53.8%	1.15	-60.4%
OAS Portfolio	66.5%	53.8%	1.15	-60.2%
Out of Sample	-91.13%	150.34%	-0.64	-20.73%
Improved by Denoising	-40.99%	15.66%	-2.91	-3.19%
Improved by Clustering	-40.99%	16.60%	-2.74	-3.19%
Improved by Clustering + Denoising	-91.13%	168.79%	-0.57	-20.73%
WalK_Forward Backtesting	40.47%	10.39%	3.46	-35.78%
K-Fold Backtesting	59.50%	17.05%	3.22	-32.77%
CPCV Backtesting	59.50%	17.05%	3.22	-32.77%

Applying advanced techniques such as Ledoit-Wolf shrinkage, clustering, and denoising can significantly enhance portfolio performance. The combination of clustering and denoising proved to be the most effective in maintaining stability while optimizing returns in an out-of-sample testing scenario. The reason we are see the K-Fold and CPCV backtesting results similar is because we are taking the purging window very small(about 10 days), and since we have a small dataset, we cannot bigger window size.

Step 5

Table 15: Combined result for all the methods with original portfolio

Combination	Portfolio Return	Portfolio Volatility	Sharpe Ratio	Max Drawdown
Original Portfolio	272.4%	150.3%	1.78	-86.4%
Ledoit-Wolf	66.5%	53.8%	1.15	-60.4%
OAS Portfolio	66.5%	53.8%	1.15	-60.2%
Out of Sample	-91.13%	150.34%	-0.64	-20.73%
Improved by Denoising	-40.99%	15.66%	-2.91	-3.19%
Improved by Clustering	-40.99%	16.60%	-2.74	-3.19%
Improved by Clustering + Denoising	-91.13%	168.79%	-0.57	-20.73%
WalK_Forward Backtesting	40.47%	10.39%	3.46	-35.78%
K-Fold Backtesting	59.50%	17.05%	3.22	-32.77%
CPCV Backtesting	59.50%	17.05%	3.22	-32.77%

Some required questions-

1. What are the differences in performance according to these metrics for the different combinations of improvements?

   So As we can see from the above table which lists all the different metric performances from different combinations.

2. What reasons might explain these differences in performances (with specific reference to the financial products, the historical data, the math of the model dynamics, etc.).

   The main difference in these metrics from the different combinations is because all the above combinations work differently in case of a non-stationary dataset with regimes. We have already concluded that we have regimes in our dataset from the backtesting enhancement section. These regimes make clustering and backtesting to produce different results.

3. In which cases, if any, does the incremental gain in performance according to the chosen metrics justify the additional complexity and effort of the improvements?

   So, mainly we have an improvement in backtesting cases from the Sharpe ratio metric, and this is because we have used the datasets in folds to produce the average weights over these folds. Since all folds produce different weights from the MVO process and regimes are also present in the dataset, we have to average over all the folds to see the weights. And in case of increasing complexity, since we have a small dataset of 3 years, we have not complicated the process by introducing the backtesting methods.

Step 6

The major problem that we have tackled during our analysis is Estimation Error in Portfolio Optimization. The major issue we faced while dealing with portfolio optimization is that it was highly sensitive to errors in parameter estimates.

Estimation error in the covariance matrix can lead to unstable weights and poor out of sample performance (“2.6. Covariance estimation — scikit-learn 1.6.1 documentation”).

To tackle this error, we have used some advanced methods CREM, Ledoit-Wolf Shrinkage, and OAS shrinkage. Let’s discuss our challenge:

Estimation error is the deviation of an estimated parameter (eg, Covariance matrix $\widehat{\Sigma}$) from its true value Σ :

$$\epsilon = \widehat{\Sigma} - \Sigma$$

Impact of portfolio optimization:

In the Markowitz framework, the optimization problem is:

$$min_w \quad w^T \Sigma w$$

$$\text{S.t. } w^T \mu = \bar{r}, w^T 1 = 1$$

So the error in $\widehat{\Sigma}$ can result in extreme or unstable weights, affecting risk, return, and drawdowns, we have seen the same in our analysis as the original portfolio exhibited high volatility (150.34%) and extreme max drawdown (–86.46%), indicating significant estimation error.

The 3 major methods that we have used are CREM, LW Shrinkage, and OAS Shrinkage, let's look into it one by one:

CREM :

- Eigenvalue decomposition:
  - $\hat{\Sigma} = V\Lambda V^T,\ \Lambda = diag(\lambda_1, \ldots.., \lambda_n)$
- Eigenvalue adjustment:
  - $\overline{\lambda}_i = \lambda_i, \lambda_i > \lambda_c\ OR\ \lambda_c, \lambda_i \le \lambda_c$
- Reconstruction:
  - $\overline{\Sigma} = V\overline{\Lambda}V^T$

Ledoit-Wolf (LW) Shrinkage Estimator:

- $\hat{\Sigma}_{LW} = \lambda I + (1 - \lambda)\hat{\Sigma}$
  - I: Target Matrix
  - λ: Shrinkage Intensity

Oracle Approximating Shrinkage (OAS) Estimator:

- $\hat{\Sigma}_{OAS} = \lambda_{OAS} I + (1 - \lambda_{OAS})\hat{\Sigma}$

Using these methods we have produced portfolios with reduced variance and improved risk measures.

<u>Data analysis and impact on portfolio metrics:</u>

Original Portfolio:

- Return: 242.83%, Variance: 150.34%, Sharpe: 1.60, Max Drawdown: –86.46%

LW & OAS Portfolio:

- Return: ~66.53%, Variance: ~53.8%, Sharpe: ~1.15, Max Drawdown: ~–60.3%

Denoised Portfolio (via CREM):

- Return: 65.26%, Variance: 28.87%, Sharpe: 1.13, Max Drawdown: –59.97%

What we have observed is the reduction in variance and drawdown, which indicates that addressing estimation error improves portfolio stability.

We have also achieved cleaner weight allocations which contribute to better risk/reward profiles. By reducing estimation error, our advanced techniques directly led to more robust portfolios that are less sensitive to market noise.

Our analysis concludes that estimation error is a major challenge in portfolio optimization, causing extreme weights and volatile portfolios. To tackle that, we used advanced methods CREM, LW, OAS, which significantly mitigate estimation error. In this CREM, adjusts eigenvalues to filter noise, LW & OAS regularize the covariance matrix, balancing bias and variance.

At last addressing estimation error through these quantitative methods provides a systematic framework for improving portfolio performance, ensuring more robust and resilient asset allocations in volatile markets.

References

1. de Prado, Marcos Lopez. *Machine Learning for Asset Managers*. Cambridge University Press, 2020.
2. “LedoitWolf — scikit-learn 1.6.1 documentation.” *Scikit-learn*, https://scikit-learn.org/stable/modules/generated/sklearn.covariance.LedoitWolf.html#sklearn.covariance.LedoitWolf. Accessed 9 March 2025.
3. “OAS — scikit-learn 1.6.1 documentation.” *Scikit-learn*, https://scikit-learn.org/stable/modules/generated/sklearn.covariance.OAS.html#sklearn.covariance.OAS. Accessed 9 March 2025.
4. “2.6. Covariance estimation — scikit-learn 1.6.1 documentation.” *Scikit-learn*, https://scikit-learn.org/stable/modules/covariance.html#shrunk-covariance. Accessed 9 March 2025.
5. Gubu, La, Dedi Rosadi, and Abdurakhman. "Robust Mean–Variance Portfolio Selection using Cluster Analysis: A Comparison between Kamila and Weighted K-Mean Clustering." Asian Economic and Financial Review, vol. 10, no. 10, 2020, pp. 1169-1186. ProQuest, https://gm10dessj-mp03-y-https-www-proquest-com.proxy.lirn.net/scholarly-journals/robust-mean-variance-portfolio-selection-using/docview/2591389266/se-2, doi:https://doi.org/10.18488/journal.aefr.2020.1010.1169.1186.
6. Goudarzi, Siamak, Mohammad J. Jafari, and Amir Afsar. "A Hybrid Model for Portfolio Optimization Based on Stock Clustering and Different Investment Strategies." International Journal of Economics and Financial Issues, vol. 7, no. 3, 2017, pp. 602-608. ProQuest,

https://gm10dessl-mp03-y-https-www-proquest-com.proxy.lirn.net/scholarly-journals/hybrid-model-portfolio-optimization-based-on/docview/2270060242/se-2.

7. Ding, Yuan, et al. "Multi-Objective Portfolio Optimization in Stock Market." International Journal of Design, Analysis and Tools for Integrated Circuits and Systems, vol. 6, no. 1, 2017, pp. 63-67. ProQuest, https://gm10desss-mp03-y-https-www-proquest-com.proxy.lirn.net/scholarly-journals/multi-objective-portfolio-optimization-stock/docview/2088783903/se-2.