Copenhagen Business School
Copenhagen, May 15, 2020

Exploration of Hierarchical Clustering in Long-Only Risk-Based Portfolio Optimization

Master Thesis

Authors:
Daniel Sjöstrand
Nima Behnejad

Supervisor:
Martin Richter

Master of Economics and Business Administration
Finance & Investments

103 Pages | 189,481 characters

Abstract

Modern portfolio optimization methods have introduced new ways of allocating capital and have drawn the attention of scholars, practitioners, and the general public alike. The thesis aims to add to the empirical evidence on the impact and risk-based performance of hierarchical clustering portfolios in long-only risk-based portfolio optimization. This is achieved by analyzing and investigating the Hierarchical Risk Parity, Hierarchical Equal Risk Contribution, and Nested Clustered Optimization methods, and compare these from a risk-based perspective to several traditional optimization methods. The relative risk-based performance is assessed through Monte Carlo simulations using synthetic data as well as through a walk-forward backtest applied on historical S&P 500 data. Together, the methodology provides a broad view of the general performance, but also more focused insights into potential estimation error reduction and the impact of different clustering parameters. The combined empirical results do not provide conclusive support for any general performance gains from hierarchical clustering in portfolio optimization. The initial positive effects found in earlier studies for Nested Clustered Optimization are hypothesized to stem from the highly stylized and simplified assumptions applied. The results given in this thesis suggest that these initial positive effects diminish when applied to more realistic data. Furthermore, the results for Hierarchical Risk Parity and Hierarchical Equal Risk Contribution show results in line with previous studies by Raffinot (2018). It is concluded that they are performing reasonably well but underperform in comparison to several of the traditional portfolios on most risk-based performance dimensions included. The findings do not indicate any general increase in risk-based performance, but do, however, show promise in providing more control over the weight concentration. In conclusion, the authors find that clustering indicates some promising aspects, but that these are limited given the applied hierarchical methodology, and further research is warranted to reach more conclusive answers.

Keywords – Asset Allocation, Portfolio Optimization, Hierarchical Clustering, Machine Learning, Risk-Based Strategies

Contents

List of Figures

List of Tables

1 Introduction

Risk management is an ever-present and highly important topic for finance practitioners and academics alike. In portfolio optimization, the question of risk and diversification is fundamental as to how portfolios are formed and evaluated.

Current portfolio optimization techniques are based on the seminal work of Markowitz (1952) with the introduction of the mean-variance optimization framework. Instead of only focusing on the returns, Markowitz (1952) instead stated that the optimization problem as a function of the covariance structure and expected returns of the asset universe. Due to practical challenges of forecasting returns, many practitioners have turned towards purely risk-based portfolio allocation methods that solely rely on the covariance matrix as an input. Some of the notable and more popular risk-based methods include the minimum-variance (Clarke, De Silva, & Thorley, 2006), maximum diversification (Choueifaty & Coignard, 2008), and equal risk contribution (Maillard, Teiletche, & Roncalli, 2008). However, the covariance matrix estimation exhibits some practicial challenges and are by no means perfect. Portfolios dependent on estimates of risk and dependencies between assets have been shown to suffer from an adverse effect of allocating larger weights to more extreme and often badly estimated components, making the optimization an "estimation-error maximization" (Michaud, 1989).

DeMiguel, Garlappi, and Uppal (2009) show that optimized portfolios often underperform even the most naive solution of equal weighting. This due to the estimation error of the covariance matrix having a larger adverse effect on the performance compared to the positive effect incurred from the portfolio optimization itself. While covariance matrix estimation error is a well-studied subject, it is still an issue in portfolio optimization with many mitigations, but no real universal solution. These mitigations can be divided into two separate regions of research. The first approach is related to the development of a more robust covariance estimation, with many proposed solutions, from the well-established shrinkage estimator first introduced by Ledoit and Wolf (2000), to less covered solutions from the field of Random Matrix Theory. The second approach is related to ways to circumvent the covariance matrix inversion and reduce estimation errors using clustering methods to find natural groupings in the data, highlighting common characteristics within groupings that are not present outside them. Hierarchical clustering represents a subset of these methods popular in finance that not only find groupings, but also their inherent hierarchical structures, providing rich information of the structures of the underlying data.

Both fields of research mentioned above, provide different ways in how one can find better estimates of the covariance matrix. As proposed by Papenbrock (2011), clustering also presents a robust method to improve performance by finding groups of assets that show collective behavior, increasing the possibility to diversify by allocating according to the groupings. He further argues that, while there are many methods within finance to categorize groups of assets, an explorative machine learning method like clustering has the benefit of not requiring any economic, causal, institutional, or psychological explanation, which are all subject to potentially erroneous assumptions. The purpose of this thesis is to further investigate this suggested impact of hierarchical clustering in portfolio optimization and add to the existing body of empirical evidence.

Many of the more recent studies on the subject of hierarchical clustering in portfolio optimization has been narrowing in on risk parity, from the Hierarchical Risk Parity (HRP) by López de Prado (2016) to Hierarchical Equal Risk Contribution (HERC) by Raffinot (2018). However, as shown in several studies, the risk parity approach is more robust to estimation error as compared to both minimum variance and maximum diversification (Ardia, Bolliger, Boudt, & Gagnon-Fleury, 2017; Jain & Jain, 2019; Zakamulin, 2015). Until recently, no empirical studies provide grounds that this problem could apply to methods more sensitive to estimation errors. López de Prado (2019a) introduced a novel way to apply clustering-based allocation in the general case, and focuses on the case of the mean-variance framework, with examples using the global minimum-variance portfolio and the maximum Sharpe ratio portfolio. This framework, referred to as Nested Clustered Optimization (NCO), allow for new possibilities in applying hierarchical clustering as a general tool in portfolio optimization instead of as a separate strategy.

1.1 Problem Discussion

Hierarchical clustering-based asset allocation strategies is a relatively new area of research, and consequently, the available literature on the subject is sparse. The Hierarchical Risk Parity method proposed by López de Prado (2016) introduced a new way to apply hierarchical clustering and has since been followed, among others, by Hierarchical Equal Risk Contribution by Raffinot (2018), and Nested Clustered Optimization by López de Prado (2019a). These methods have been shown to provide competitive performance as separate and isolated strategies in comparison to several benchmark strategies. It has further been theorized that the NCO framework could be used to improve traditional portfolio optimization strategies by applying it as a step in the optimization process. As shown by López de Prado (2019a), the Nested Clustered Optimization method serves as a powerful tool to decrease the estimation error for the minimum-variance portfolio, and the potential for similar results using different portfolios with other optimization objectives could prove fruitful.

The goal of this thesis is to further add to the empirical evidence of clustering-based portfolio optimization performance in comparison to more traditional techniques. To achieve this, the theory and applications of long-only hierarchical clustering-based methods are applied and compared against the long-only minimum-variance, maximum diversification, and equal risk contribution portfolios. The study provides an analysis and comparison of their relative risk-based performances using Monte Carlo simulations together with a historical backtest using data on the S&P 500. The research question of this thesis can thus be formulated as:

- How do hierarchical clustering-based portfolios perform from a risk-based perspective in comparison to each other and to their non-hierarchical counterparts?

To answer the above-stated research question, several sub-questions are posed to further examine the possible drivers of the performance, as well as the practical applicability of the methods:

- Can the estimation error of the traditional portfolios be reduced using Nested Clustered Optimization?
- How do different linkage methods affect the portfolio optimization results?
- How well do the portfolios perform from a risk-based perspective?
- How concentrated are the weights of the different portfolios?
- What level of asset turnover is incurred by the different portfolios?

With these questions, the thesis intends to gather practical and theoretical insights, as well as to extend the empirical evidence of the effects of said allocation strategies. These insights are expected to fill the void as to how the positive effects from hierarchical clustering generalize from the global minimum-variance portfolio to several traditional long-only portfolio strategies. In addition, the empirical results aspire to provide further grounds for investors planning to apply hierarchical-clustering based asset allocation strategies to their arsenal.

1.2 Delimitations

Several delimitations have been imposed in this thesis to create comparable results and to retain focus on the central questions of the problems presented.

When referring to clustering, only hierarchical clustering is included. Several different varieties of clustering algorithms exist, and original implementations for the NCO portfolios is originally implemented using a similar but different method. For the sake of comparability, all clustering-based portfolios utilize the same algorithm, namely the agglomerative hierarchical clustering algorithm. Some of the cluster-based portfolios included in the thesis are thus slightly modified to highlight the differences between the weight allocation methods, and not the clustering quality itself. The risk-based traditional portfolio optimization methods in this thesis are limited to the long-only variants of the minimum variance portfolio, equal risk contribution portfolio, and the maximum diversification portfolio.

When comparing the risk-based performance and return characteristics of the different portfolios, the point estimates are used, and no confidence intervals indicating formal significance is applied. In addition, for all computational methods and models, returns are assumed to be independent and identically distributed (iid). Results provided are not correct for, nor regard for any violations of these assumptions.

The investment universe used in the thesis utilizes the S&P 500 equity inverse in the U.S. equities market between January 2nd, 1990 and January 17th, 2020. All price data is comprised of constituents included in this index, and any other equity or investment asset outside this scope is not taken into consideration. Besides, once the constituents is included in the asset universe, it is only removed once it no longer fulfills the criteria, regardless of the inclusion or exclusion decision by the S&P 500 itself. Also, the study does not consider any effects of market imperfections such as transaction costs nor is any external tax implications considered.

1.3 Structure

The thesis is divided into five sections, with the first section concluded above. The four following sections of the thesis are summarized below.

Section 2 – Theory

In this section, the theoretical framework is presented. It starts by going through the assumptions and mathematical definitions. Following this, estimation instability of the covariance and correlation matrix is explored to provide an understanding of estimation errors and its different underlying sources. Next, the theory and practical application of shrinkage estimation and hierarchical clustering are introduced as different solutions to mitigate the before mentioned estimation errors. Finally, the portfolio optimization techniques are presented and discussed, starting with the more traditional methods such as the MV, MD, and ERC portfolios. Following this, there is a more in-depth presentation of the hierarchical clustering-based techniques, including HRP, HERC, and NCO.

Section 3 – Data

The data section aims to provide the reader with a detailed view of the data used in this thesis. First, the applied sample selection is presented, including the S&P 500 investment universe and detailed information about the index. Secondly, data collection and data processing methodology is explored. Thirdly, the proxy for the risk-free rate is presented, and last, the synthetic data generation is explained to provide the necessary background to how the Monte Carlo simulations are carried out.

Section 4 – Methodology

In this section, the overarching research methodology is presented. First, the definitions for returns, volatility, covariance estimation, and shrinkage estimation are presented. Second, the implementation of the agglomerative hierarchical clustering algorithm is introduced, providing the practical implementation of the first step of the HRP, HERC, and NCO portfolios. Third, the practical application and implementation of all portfolios are explained. Finally, the backtesting methodology is presented and discussed, together with the applied statistical performance metrics.

Section 5 – Empirical Results

In this section, the empirical results are presented following the backtesting methodology presented in section 4. First, an initial Monte Carlo simulation is carried out, following the methodology as presented by López de Prado (2019a) to investigate the proposed reduced estimation error from applying NCO to portfolio optimization. Second, a more extensive Monte Carlo simulation is carried out to investigate the in-sample to out-of-sample performances of the different portfolios, with an emphasis on the choice of linkage method used in the clustering-based portfolios. Finally, the results from the walk-forward historical backtest using S&P 500 data ranging from 1992-01-02 to 2020-01-17 is presented.

Section 6 – Discussion

In this section, the results of the empirical testing are discussed, focusing on the interpretations of the empirical findings and their similarities and contrasts to results from prior studies and in connection to the theoretical framework presented in section 2. Following this, the implications of the practical applicability and areas of potential further research are presented.

Section 7 – Conclusion

Finally, the thesis is concluded by providing answers and interpretations of the research question along with the sub-questions.

2 Theory

This section introduces the theoretical framework used in this thesis. First, the assumptions and definitions of the theory are presented, followed by stylized facts observed in the correlation matrix. Secondly, the central problem and potential remedies of estimation errors in the covariance matrix is discussed. Third and finally, the traditional and hierarchical-clustering based portfolios are presented.

2.1 Assumptions

All assets included in the asset universe are assumed to be perfectly divisible and perfectly liquid, where all stocks have willing buyers to pay the closing price, and all shares have a willing seller who offers to sell at the closing price at any given time. Also, markets are assumed to be frictionless, where no transaction costs are incurred either from buying or selling assets. Tax effects on asset returns and dividends are excluded, assuming that before-tax and after-tax returns are equal for all investors.

The study restricts portfolio allocations to be long-only, meaning that no negative weights (or differently put, short-sales) are allowed and that the total weight of each portfolio adds to one. It is further assumed that the investor always reinvests any dividend from an asset back into the same asset.

2.2 Definitions

For the sake of clarity, definitions concerning mathematical notation and necessary calculations for the derivation of portfolios are presented. Matrices are presented using bold-face upper-case letters (e.g. $\boldsymbol{M}$), and vectors using bold-face lower-case letters (e.g. $\boldsymbol{v}$).

The investment universe consists of n assets that the investor can allocate weight to at any time t, $t = 0, 1, \ldots, T$. The prices of these assets are given by

$$P_{i,t} \quad \text{for} \quad i = 1, \ldots, n \quad \in \quad t = [0, T]. \tag{2.1}$$

The *net return*, or simple return, on the asset over the time period $t-1$ to t is defined as

$$r_{i,t} = \frac{P_t - P_{t-1}}{P_{t-1}} = \frac{P_t}{P_{t-1}} - 1. \tag{2.2}$$

Thus, the return on any asset is simply the proportional change of its price over a given time period. The vector of n rates of return is defined as

$$\boldsymbol{r}_t = (r_{1,t}, \ldots, r_{n,t})^\top. \tag{2.3}$$

The average return of asset i over the period t to T can be defined as

$$\bar{r}_i = \frac{1}{T}\sum_{t=1}^{T} r_{it}. \tag{2.4}$$

The *geometric mean* is usually preferred over the *arithmetic mean* given above when measuring the performance as rate of returns are multiplicative over time rather than additive

$$\mu_i = \left(\prod_{t=1}^{T}(1 + r_{it})\right)^{\frac{1}{T}} - 1. \tag{2.5}$$

The second moment of the return distribution of asset i, the variance up until time T is defined as

$$\sigma_i^2 = Var(r_i) = \frac{1}{T}\sum_{t=1}^{T}(r_{it} - \bar{r}_i)^2. \tag{2.6}$$

The *covariance function* gives the covariance of two assets, that is, the co-movement of the two assets over time. The covariance of the returns of asset i and j are defined as

$$\rho_{ij} = \text{cov}(r_i, r_j) = \frac{1}{T}\sum_{t=1}^{T}(r_i - \bar{r}_i)(r_j - \bar{r}_j). \tag{2.7}$$

Hence, the covariance of an asset with itself $i = j$ for any t is simply

$$\text{cov}(r_i, r_i) = \sigma_i^2. \tag{2.8}$$

The correlation function is the standardized and demeaned covariance function between asset i and asset j,

$$\rho_{ij} = \text{corr}(r_i, r_j) = \frac{\sigma_{ij}}{\sigma_i \sigma_j}. \tag{2.9}$$

By definition, the correlation can vary from perfectly negative (-1) to perfectly positive (1), and where the assets are fully uncorrelated when $\rho_{ij} = 0$. The portfolio returns at time t are defined as the relative weighted returns, formally stated as

$$r_P = \sum_{i=1}^{n} w_i r_i = \boldsymbol{w}^\top \boldsymbol{r}, \tag{2.10}$$

where the vector of all portfolio weights can be defined as

$$\boldsymbol{w} = (w_1, \ldots, w_n)^\top. \tag{2.11}$$

The sum of all weights are stated as a fraction of the entire portfolio, and are required to sum to one

$$\boldsymbol{w}^\top \mathbf{1} = \sum_{i=1}^{n} w_i = 1. \tag{2.12}$$

The portfolio variance, or the weights variance can be calculated using

$$\sigma_P^2 = \sum_{i,j} \rho_{ij}\sigma_i\sigma_j w_i w_j = \boldsymbol{w}^\top \boldsymbol{\Sigma} \boldsymbol{w}, \tag{2.13}$$

where $\boldsymbol{\Sigma}$ represents the population covariance matrix. Since the true population covariance is not directly observable, it can be estimated from the sample covariance matrix

$$\hat{\boldsymbol{\Sigma}} = T^{-1} \boldsymbol{X} \left(\boldsymbol{I} - T^{-1} \mathbf{1}\mathbf{1}^\top \right) \boldsymbol{X}^T, \tag{2.14}$$

where $\boldsymbol{X}$ denotes an $N \times T$ matrix of T returns on a universe of N stocks. Using matrix notation, the sample covariance matrix can be expressed as

$$\hat{\boldsymbol{\Sigma}} = \begin{pmatrix} \sigma_{11}^2 & \sigma_{12} & \ldots & \sigma_{1n} \\ \sigma_{21} & \ddots & \ldots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \sigma_{n1} & \ldots & \ldots & \sigma_{nn}^2 \end{pmatrix}. \tag{2.15}$$

The sample correlation matrix $\hat{\boldsymbol{C}}$ can be constructed by centering and normalizing the sample covariance matrix

$$\hat{\boldsymbol{C}} = \mathrm{diag}(\boldsymbol{\Sigma})^{-\frac{1}{2}} \boldsymbol{\Sigma}, \mathrm{diag}(\boldsymbol{\Sigma})^{-\frac{1}{2}} \tag{2.16}$$

where $\mathrm{diag}(\boldsymbol{\Sigma})$ denotes the diagonal of the covariance matrix, that is, the variances. The correlation matrix is a symmetric matrix with $\rho_{ii} = 1$ in the main diagonal, and the respective pairwise correlation in the off-diagonal values, stated in matrix form as

$$\hat{\boldsymbol{C}} = \begin{pmatrix} 1 & \rho_{12} & \dots & \rho_{1n} \\ \rho_{21} & \ddots & \dots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ \rho_{n1} & \dots & \dots & 1 \end{pmatrix}. \tag{2.17}$$

The eigenvalues and eigenvectors of the financial correlation and covariance matrix are of central importance when discussing the implications of clustering-based portfolios. The eigenvectors represent linear combination of orthogonal vectors explaining the maximum amount of variance in the data, ordered in terms of the absolute size of their respective eigenvalues, representing the explanatory power of the eigenvector (Pasini, 2017). The mathematical derivation of eigenvectors and eigenvalues is not of importance in this paper. However, an understanding of their properties and applications will make the following sections easier to follow.

In essence, an eigenvector, $\boldsymbol{v}$, represents a vector that is unaffected by a linear transformation T. Applying the linear combination will thus not change the direction, only the magnitude, referred to as the eigenvalue, λ. More formally, this can be expressed as

$$T(\boldsymbol{v}) = \lambda \boldsymbol{v}. \tag{2.18}$$

The method of decomposing a matrix into it's eigenvectors and eigenvalues is sometimes also referred to as the Principal Component Analysis (PCA). This is often used in finance to derive and approximate underlying factors exposures, implying that a common exposure to an eigenvector corresponds to a common factor exposure (Pasini, 2017).

2.3 Stylized Facts of the Financial Correlation Matrix

In this section, the stylized facts of sample correlation matrices for financial asset returns are discussed to build intuition around the possible ways to leverage the information contained in the matrix to its fullest and to set up guides on which to compare the synthetic correlation matrices to their empirically observed counterparts at a later stage of the study. The discussed list of stylized facts follows the list outlined by Marti (2019).

Stylized Fact 1: Positive Pairwise Correlation

The distribution of pairwise correlations between assets is significantly positively shifted. In other words, the mean correlation is significantly positive (Marti, 2019).

Stylized Fact 2: Marčenko-Pastur Distributed Eigenvalues

Eigenvalues of the empirical correlation matrix are shown to follow the Marčenko-Pastur distribution but with a very large first eigenvalue that represents the market mode, and a couple of other large eigenvalues, proposed to represent industries (Laloux, Cizeau, Potters, & Bouchaud, 2000).

Stylized Fact 3: Perron-Frobenius Property

In the general case, financial correlation matrices have a first eigenvector (defined as the eigenvector with the largest eigenvalue) with only positive entries. This is in line with the Perron-Frobenius theorem which asserts that a real square matrix with positive entries has a largest real eigenvalue with a corresponding eigenvector that have strictly positive components as shown by Perron (1907) and Frobenius (1912).

Stylized Fact 4: Hierarchical Structure of Correlations

Mantegna (1999) provides evidence of a hierarchical structure in the S&P 500 index correlation coefficient matrix. With only a few exceptions, the groupings discovered were homogeneous with regards to industry and often also sub-industry sectors implied by GICS. The detection of this hierarchical structure in a broad portfolio of stocks traded in a financial market has several implications. First, different stocks are affected by different factors, and to varying degrees. Second, in general, groups of stocks that depart early in a hierarchical clustering tree (at higher values of dissimilarity) are relatively more affected by economic factors specific to that group. Conversely, when the departure is lower down in the tree, the stocks are more likely to be affected by economic factors common to all stocks or other groups.

Stylized Fact 5: Scale-Free Property of the Corresponding Minimum Spanning Tree

The minimum spanning tree (MST) network formed by the correlation matrix exhibit a scale-free property, defined by the distribution of the number of edges of the nodes, $P(k)$ follows a so-called power-law distribution

$$P(k) \sim k^{-\gamma} \tag{2.19}$$

This stylized fact is widely accepted and has been empirically shown in several studies, from the initial paper by Vandewalle, Brisbois, Tordoir, et al. (2001) over a short time horizon, to more robust findings by Kim, Kim, Lee, and Kahng (2002), and Bonanno, Caldarelli, Lillo, and Mantegna (2003).

2.4 Estimation Instability

In this section, the estimation instability of the sample covariance and correlation matrix is discussed to provide necessary background to the problem, together with its different underlying sources. With this intuition, potential remedies can later be presented and discussed in more detail.

Many portfolio optimization techniques rely on the sample covariance matrix for the weight allocation, and the quality of the input covariance matrix has large significance on the quality of the weights produced to align with the optimization objective. This has important implications as more accurate estimations produce weight allocations closer to the optimal solution. More specifically, the numerical stability of the portfolio optimization dependent on a covariance matrix is determined by the condition number, defined as the absolute value of the ratio between the maximum and minimum eigenvalues (López de Prado, 2016). When more correlated assets are added, the condition number grows, and the matrix becomes more ill-conditioned. As such, the benefits of diversification by adding more assets are often offset by the estimation errors incurred by them (López de Prado, 2016). In addition, one should carefully distinguish the *empirical* matrices $\hat{\boldsymbol{\Sigma}}$ and $\hat{\boldsymbol{C}}$ from the *true* population matrices $\boldsymbol{\Sigma}$ and $\boldsymbol{C}$, as the former only estimates the unknown and unobservable latter. With this estimation, additional noise is produced.

In conclusion, instability in estimating the covariance and correlation matrix mainly stems from *noise* due to sampling errors, but also to *signal*, much through the inherent hierarchical structure of financial cross-correlations (López de Prado, 2020). These two sources of estimation error are further examined in the two coming sub-sections.

2.4.1 Noise as a Source of Covariance Instability

In the classical statistical limit, i.e., when the number of observations $T \to \infty$ with a fixed number of assets N, the law of large numbers implies that the estimator $\hat{\boldsymbol{\Sigma}}$ converges towards the true value of population covariance matrix $\boldsymbol{\Sigma}$. However, in the high-dimensional case where the assets N and the observations T both are large, $\hat{\boldsymbol{\Sigma}}$ is considered a noisy estimator of $\boldsymbol{\Sigma}$, especially in terms of eigenvalues and eigenvectors (Bun, Bouchaud, & Potters, 2017). When not acknowledging this fact, the *in-sample*, predicted risk widely underestimates the *true* risk over the same period, and even more so the future *out-of-sample*, realized risk in the consecutive estimation period (Bun et al., 2017).

The Marčenko-Pastur distribution from Random Matrix Theory is a fundamental tool to analyze the eigenvalue spectrum of large correlation matrices in this *large-dimensional limit*, where both the number of assets N and the number of observations T become large, but with their ratio $q = N/T$ not being vanishingly small (Bun et al., 2017). This distribution can help to infer the relationship between the empirical correlation matrix $\hat{\boldsymbol{C}}$ and the real correlation matrix $\boldsymbol{C}$ in terms of the distribution of their eigenvalues. More concretely, the distribution captures the distortion of the eigenvalue spectrum of $\hat{\boldsymbol{C}}$ as an increasing function of the ratio q (Marčenko & Pastur, 1967), a phenomenon commonly known as the *curse of dimensionality.* For a correlation matrix $\boldsymbol{C}$, the Marčenko-Pastur distribution follow

$$\rho_C(\lambda) = \frac{T}{N}\frac{\sqrt{(\lambda_+ - \lambda)(\lambda - \lambda_-)}}{2\pi\lambda\sigma^2} \tag{2.20}$$

where

$$\lambda_+ = \sigma^2\left(1 + \sqrt{N/T}\right)^2 \tag{2.21}$$

$$\lambda_- = \sigma^2\left(1 - \sqrt{N/T}\right)^2 \tag{2.22}$$

The result of this distribution provides information on how the distribution of eigenvalues of a null hypothesis is affected by the ratio between assets and periods of observations, and can thus infer meaningful eigenvalues, $\lambda \in [\lambda_-, \lambda_+]$, from noisy ones, $\lambda \notin [\lambda_-, \lambda_+]$.

Figure 2.1 shows the probability density function for the null hypothesis $\boldsymbol{C} = \boldsymbol{I}_N$ where $N = 500$. This shows that even in the case where there are no correlations in the population covariance matrix, spurious eigenvalues higher than one still exist, increasing the condition number, and thus decreasing the stability of the covariance matrix. The results from the above case can be extrapolated to the case of estimating the financial covariance matrix, implying that the estimation process incur some amount of noise as an increasing function of q. Figure 2.2 depicts an observed eigenvalue distribution of U.S stock data compared to a null hypothesis of the identity matrix with the same q, illustrating the distortion of the eigenvalue spectrum from

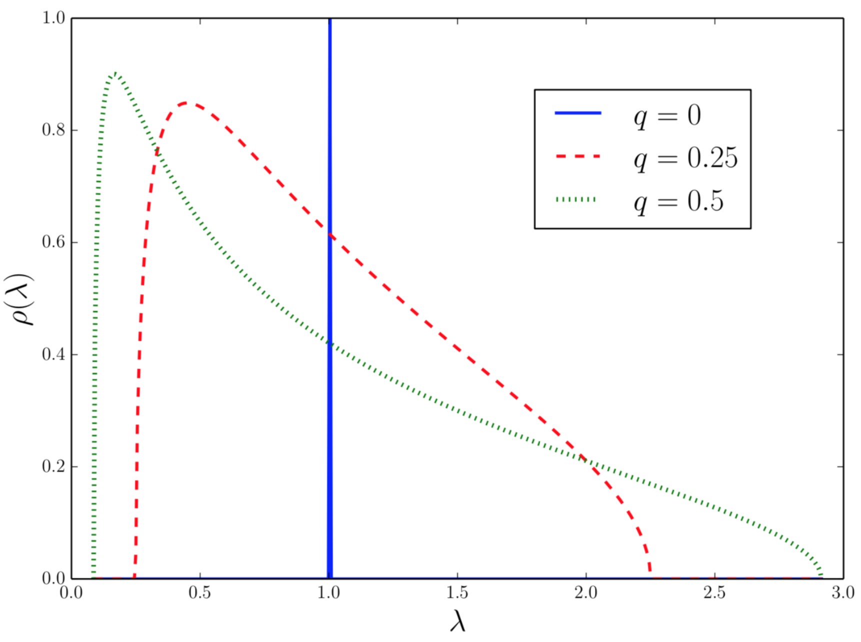

Figure 2.1: Sample eigenvalue density under the null hypothesis of $\boldsymbol{C} = \boldsymbol{I}_N$ with $N = 500$

Source: Bun, Bouchaud, and Potters (2017)

estimation errors.

With this in mind, allocation methods using the covariance matrix are in theory increasingly noisy as the portfolio size increases without a proportional response in the increase of observations used. Several different methods can be applied in practice to mitigate this estimation problem. In this thesis, Ledoit-Wolf shrinkage estimation is applied as it has been empirically shown to produce robust estimation error reduction (Ledoit & Wolf, 2004), later examined in section 2.5.

2.4.2 Signal as a Source of Covariance Instability

A different source of financial covariance instability comes from the structure of the data (López de Prado, 2020). As cross-correlation between assets affect the condition number of the matrix, regardless and independent of q, the signal-induced instability cannot be reduced by sampling more observations alone. Outside the ideal case when there is zero correlations, $\boldsymbol{C} = \boldsymbol{I}_N$, the condition number is higher than one. In the empirically observed case, non-zero correlations between assets and between groups of clusters are present, emphasizing that this is not the case (Mantegna, 1999).

When a subset of assets display a higher correlation among themselves than with the rest of

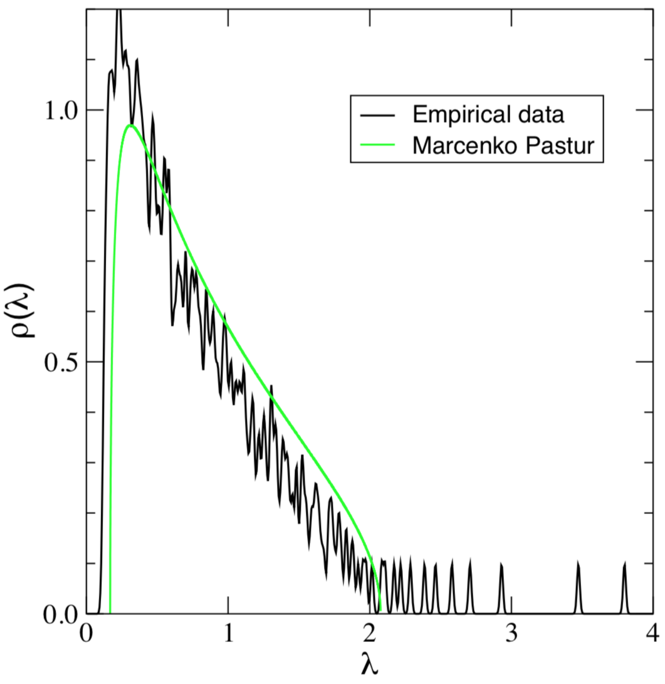

Figure 2.2: Eigenvalue distribution of U.S. stock data with $N = 406$ and $T = 1300$, compared to a null hypothesis of $\boldsymbol{C} = \boldsymbol{I}_N$.

Source: Bun, Bouchaud, and Potters (2017)

the asset universe, they can be said to form a cluster. Mantegna (1999) argues that the same common economic factors drive these subsets as first introduced by Ross (1976) in his seminal work on the *arbitrage theory of capital asset pricing.* According to this theory, different assets are driven by different economic factors to varying degrees. Mantegna (1999) provides empirical findings on how clusters between assets can be found within the correlation matrix. Assets within the same cluster are more exposed to a common eigenvector, which implies that its associated eigenvalue can explain a larger amount of the total variance of those assets. The trace of the correlation matrix, $\mathrm{tr}(\boldsymbol{C})$, is exactly N, as the diagonal values in the correlation matrix are all equal to 1. This means that one eigenvalue can only increase at the cost of the other eigenvalues as the sum of eigenvalues is constant, showed by

$$\mathrm{tr}(\mathbf{C}) = N = \sum_{i=1}^{N} \rho_{ii} = \sum_{i=1}^{N} \lambda_i. \tag{2.23}$$

This results in a higher condition number as the difference between the highest and lowest eigenvalues $[\lambda_-, \lambda_+]$ increases in both directions. The implication of this is that a greater intra-cluster correlation leads to a higher condition number. The condition number is ultimately dominated by the cluster with the highest correlation, as this is determined by the most extreme eigenvalue. López de Prado (2020) suggests mitigating this problem by disassembling the covariance matrix into smaller covariance matrices and optimize these separately to contain the instabilities using clustering methods, further explored in section 2.6 and section 2.10.

2.5 Shrinkage Estimation

With the theory from section 2.4 in mind, the empirical covariance and correlation matrix should be "cleaned" to decrease the estimation error of true and out-of-sample risk to a high degree as possible, and thus, decrease the probability of the optimization to spuriously respond to noise. A common way to mitigate these problems is by utilizing a Bayesian method referred to as a shrinkage estimation that assumes a prior covariance matrix $\boldsymbol{F}$, and shifts the observed matrix $\boldsymbol{S}$ towards it

$$\hat{\boldsymbol{\Sigma}} = \delta \boldsymbol{F} + (1-\delta)\boldsymbol{S}. \tag{2.24}$$

The shrinkage intensity is determined by the shrinkage coefficient parameter $\delta \in [0,1]$. This method can be applied to increase the accuracy of the inverse covariance matrix by reducing the condition number (Ledoit & Wolf, 2004). Shrinkage estimation involves the execution of two steps. First, the shrinkage target $\boldsymbol{F}$ is determined, and second, the shrinkage coefficient δ is estimated (Ledoit & Wolf, 2004).

2.5.1 Shrinkage Target

The shrinkage target $\boldsymbol{F}$ follows that of Ledoit and Wolf (2004), where a *constant correlation model* is used, defined as the average pairwise correlation in the covariance matrix, and thus assumes that all pairwise correlations are identical. This choice of shrinkage target exploits the stylized fact that pairwise correlations coefficients between financial assets are significantly positively shifted, also shown in section 2.3. The average sample correlations are given by

$$\bar{\rho} = \frac{2}{(N-1)N} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \rho_{ij}. \tag{2.25}$$

From this, define the constituents f of the correlation matrix $\boldsymbol{F}$ as the average sample correlation multiplied with the sample covariance for the off-diagonal elements

$$f_{ii} = s_{ii} \quad \text{and} \quad f_{ij} = \bar{\rho}\sqrt{s_{ii}s_{jj}} \tag{2.26}$$

Thus, the shrinkage target covariance matrix can be expressed in matrix form as

$$\boldsymbol{F} = \begin{pmatrix} s_{11} & f_{12} & \dots & f_{1n} \\ f_{21} & \ddots & \dots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ f_{n1} & \dots & \dots & s_{nn} \end{pmatrix}. \tag{2.27}$$

2.5.2 Shrinkage Constant

Choosing the amount of shrinkage, δ amounts to balancing the bias/variance trade-off. Ledoit and Wolf (2004) propose a formula to compute the optimal shrinkage coefficient δ^* that minimizes the expected distance between the shrinkage estimator and the true covariance matrix (Ledoit & Wolf, 2004). This is based on a quadratic measure of distance between the true and the estimated covariance matrices based on the Frobenius norm, which for a symmetric $N \times N$ matrix Z can be defined as

$$\|Z\|^2 = \sum_{i=1}^{N}\sum_{j=1}^{N} z_{ij}^2. \tag{2.28}$$

The Frobenius norm is used to compute the difference between the shrinkage estimator and the true covariance matrix, following a quadratic loss function, defined as

$$L(\delta) = \|\delta + \boldsymbol{F} + (1-\delta)\boldsymbol{S} - \boldsymbol{\Sigma}\|^2. \tag{2.29}$$

Under the assumption that N is fixed while T moves to infinity, the optimal asymptotic value of δ^* behaves as a constant over T. This constant κ is defined as

$$\kappa = \frac{\pi - \rho}{\gamma} \tag{2.30}$$

In practice, κ is unknown, and consistent estimators for it has to be found by locating consistent estimators for its three variables π, ρ, and γ. First, a consistent estimator for π is

$$\hat{\pi} = \sum_{i=1}^{N}\sum_{j=1}^{N} \hat{\pi}_{ij} \quad \text{with} \quad \hat{\pi}_{ij} = \frac{1}{T}\sum_{t=1}^{T} \left\{ (y_{it} - \bar{y}_i)(y_{jt} - \bar{y}_j.) - s_{ij} \right\}^2. \tag{2.31}$$

Second, an estimator for ρ can be defined as

$$\hat{\rho} = \sum_{i=1}^{N} \hat{\pi}_{ii} + \sum_{i=1}^{N}\sum_{j=1, j\neq i}^{N} \frac{\bar{r}}{2}\left(\sqrt{\frac{s_{jj}}{s_{ii}}}\hat{\vartheta}_{ii,ij} + \sqrt{\frac{s_{ii}}{s_{jj}}}\hat{\vartheta}_{jj,ij} \right). \tag{2.32}$$

Third, a consistent estimator for γ is

$$\hat{\gamma} = \sum_{i=1}^{N} \sum_{j=1}^{N} (f_{ij} - s_{ij})^2 \tag{2.33}$$

Putting everything together, the estimator of κ can be derived

$$\hat{\kappa} = \frac{\hat{\pi} - \hat{\rho}}{\hat{\gamma}}. \tag{2.34}$$

The estimated shrinkage intensity is then finally computed as

$$\hat{\delta}^* = \max\left\{0, \min\left\{\frac{\hat{\kappa}}{T}, 1\right\}\right\}. \tag{2.35}$$

Through the above definition of the shrinkage target and the shrinkage constant, the operational shrinkage estimator of the covariance matrix $\mathbf{\Sigma}$ is defined as

$$\hat{\mathbf{\Sigma}}_{\text{Shrink}} = \hat{\delta}^* \boldsymbol{F} + \left(1 - \hat{\delta}^*\right) \boldsymbol{S} \tag{2.36}$$

2.6 Hierarchical Clustering

In this section, hierarchical clustering is introduced to provide intuition about its methodology, what problem it is proposed to solve, and how it is applied to the hierarchical-clustering portfolio optimization problem.

Cluster analysis aims to find natural groupings in the data where the subsets can be said to be more closely related to one another than to objects assigned to other clusters (Hastie, Tibshirani, & Friedman, 2009). This type of problem appears naturally in finance, throughout many of the steps in the investment process. Existing literature on clustering in portfolio optimization often categorizes assets by utilizing clustering and uses the factors to create a portfolio (Papenbrock, 2011). Examples of this include Zhang and Maringer (2009), who present a clustering criterion that groups assets together to maximize the Sharpe ratio of the portfolio, and Dose and Cincotti (2005), limiting the number of assets included in the portfolio by considering a subset from each cluster and then setting the weight according to an optimization process. Other authors use clustering as a means of filtering and improving parameter estimation, which can be considered a filtering procedure in which the complexity of the correlation matrix is reduced (Papenbrock, 2011). Furthermore, the author found that portfolio optimization methods applied through hierarchical clustering comes with its advantages and disadvantages. For example, Papenbrock (2011) found that the method provides superior risk-adjusted performance than classic allocation strategies tested. However, he also find that the portfolio turnover and to some extent the portfolio concentration for these types of allocation methods seem to increase in comparison to non-hierarchical methods.

In this study, clustering is used to find and group financial assets within the investment universe based on a distance derived from the sample correlation matrix. These clusterings are proposed to reduce the signal-induced covariance instability by reducing the weight allocation task to several smaller sub-tasks made on smaller and more stable covariance matrices (López de Prado, 2020). Also, it is hypothesized that the cluster-based weight allocation can better account for the hierarchical structures present in empirical correlation matrices on the S&P 500, as shown by Mantegna (1999). While there exist several types of clustering algorithms, the focus on this study is on hierarchical clustering due to its broad application in previous studies (López de Prado, 2016; Raffinot, 2017, 2018).

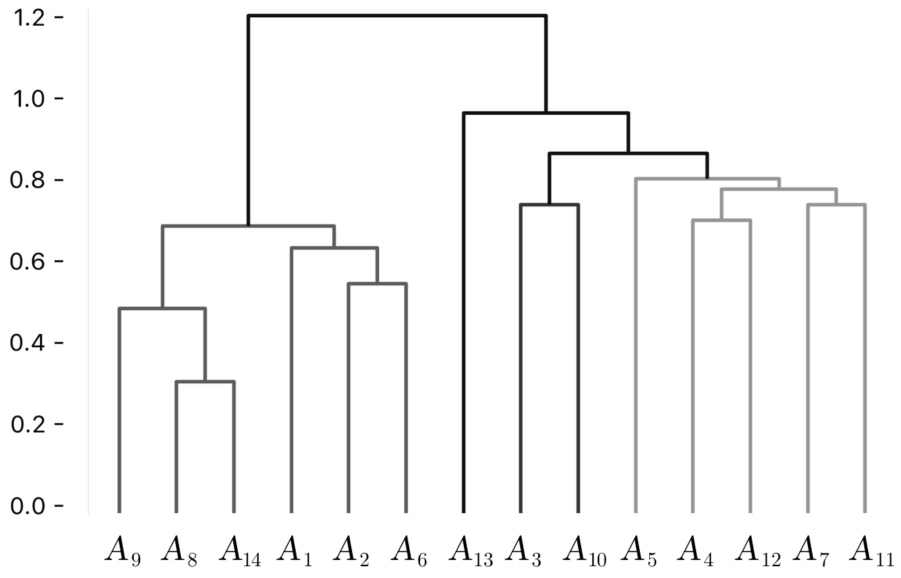
1.2
1.0
0.8
0.6
0.4
0.2
0.0
A_9 A_8 A_{14} A_1 A_2 A_6 A_{13} A_3 A_{10} A_5 A_4 A_{12} A_7 A_{11}

Figure 2.3: Dendrogram depicting the resulting binary tree from a hierarchical clustering

Hierarchical clustering produces hierarchical representations by either merging clusters create clusters at each level of the hierarchy at a lower level (agglomerative method) or dividing clusters at a higher level (divisive method). Regardless of the method used, hierarchical clustering uses the dissimilarity between groups of observations, defined through a dissimilarity metric. The dissimilarity metric is the measure of the distance between two entities, usually measured in

Euclidean terms through a *distance matrix*. The graphical representation of hierarchical clustering can be presented using a dendrogram (Figure 2.3), which provides a complete description of the clustering. This highly interpretable visualization is considered one of the main reasons for the popularity of the method (Hastie et al., 2009).

2.6.1 Correlation Matrix to Distance Matrix

A distance matrix represents the pairwise distance between objects N based on one or several similarity or dissimilarity features. The correlation coefficient matrix of asset returns is not directly applicable to use as a distance metrics as is does not fulfill the three axioms that define a Euclidean metric (Kraskov, Stögbauer, & Grassberger, 2004). A generalized metric proposed by Gower (1966) can be defined to transform the pairwise correlations, ρ_{ij}, into a distance metric d_{ij}.

$$d_{ij} = \sqrt{\frac{1}{2}\left(1 - \rho_{ij}\right)} \tag{2.37}$$

This metric fulfills the aforementioned axioms; that (i) the distance is zero only if the stochastic process or stochastic processes are exactly aligned, $d_{ij} = 0$ if $i = j$; (ii) distances are symmetric as the correlation coefficient matrix is symmetric by definition, $d_{ij} = d_{ji}$; and (iii) the triangular inequality, that the distance between two points in space d_{ij} is the shortest path between point i and point j, $d_{ij} \leq d_{ik} + d_{kj}$. The last axiom is extensively empirically observed by Mantegna (1999), whereas the first two axioms are true by definition.

The sample correlation matrix $\hat{\boldsymbol{C}}$ can thus be converted into a distance matrix $\boldsymbol{D}$ using the pairwise calculations from equation 2.37.

$$\boldsymbol{D} = \begin{pmatrix} 0 & d_{12} & \dots & d_{1n} \\ d_{21} & \ddots & \dots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ d_{n1} & \dots & \dots & 0 \end{pmatrix} \tag{2.38}$$

The Euclidean distance between the pairwise column-vectors of the distance matrix $\boldsymbol{D}$ is then computed to get the proper distance metric (López de Prado, 2016),

$$\tilde{d}_{ij} = \tilde{d}[D_i, D_j] = \sqrt{\sum_{n=1}^{N} \left(d_{ni} - d_{nj}\right)^2}. \tag{2.39}$$

The difference between the d_{ij} and $\tilde{d}_{ij}$ is subtle, where the first measures the distance of the column vectors of the correlation matrix $\hat{\boldsymbol{C}}$, and where the second measures the distance on the column vectors of $\boldsymbol{D}$, that is, a distance of distances (López de Prado, 2016).

This final distance matrix, denoted $\tilde{\boldsymbol{D}}$, is then used in the portfolio construction as a tool to find correlation clustering within the asset universe through the *agglomerative hierarchical clustering algorithm.*

2.6.2 Agglomerative Hierarchical Clustering Algorithm

Of the two possible methods implementing hierarchical clustering, the *agglomerative* hierarchical clustering algorithm is used over the *divisive* hierarchical clustering algorithm due to its broader academic base and support (Hastie et al., 2009). Agglomerative clustering begins with every observation representing a singleton cluster, that is, every observation is its own cluster. At each iteration, the two least different clusters are merged. The process continues until only one all-encompassing cluster remains. This results in the formation of a the binary tree (dendrogram), depicting the complete hierarchical structure of the entities.

The criteria of which assets or clusters to merge in each step is determined by the intergroup distance between clusters. The intergroup distance can be determined using several different procedures, referred to as *linkage* methods. Below, four common linkage methods are provided.

2.6.2.1 Single Linkage

Single Linkage defines the distance between two clusters as the shortest possible distance between them, more concretely, the distance between the two most similar entries between the clusters

$$d_{C_i,C_j} = \min_{x,y} \{D(x,y) | x \in C_i,\ y \in C_j\}. \tag{2.40}$$

Bonanno, Lillo, and Mantegna (2001) argues for the use of the Single Linkage as each element selects the most relevant connection in the set as defined by the connection with the shortest distance. In other words, it creates the $N-1$ edged hierarchical tree that minimizes the sum of the edge distances. However, it only requires a single dissimilarity $d_{ii'}$, $i \in x$ and $i' \in y$, to be small for the groups x and y to be considered close together. This can lead to a chaining effect that violates the *compactness* property, i.e., that all observations within each cluster tend to be similar to each other (Hastie et al., 2009; Raffinot, 2018).

2.6.2.2 Complete Linkage

Complete Linkage defines the distance between two clusters as the longest possible distance between their entries within, more concretely, the distance between the two least similar entries between the clusters.

$$d_{C_i,C_j} = \max_{x,y} \{D(x,y) | x \in C_i,\ y \in C_j\}. \tag{2.41}$$

Complete Linkage is at the opposite of the spectrum to a Single Linkage. In this case, two clusters are only considered close if all of the observations in the union are relatively similar. Therefore, it tends to produce compact, small clusters. This tendency violates the *closeness* property, meaning that it can produce clusters with entries that are closer to other clusters than they are to some members of their cluster (Hastie et al., 2009).

2.6.2.3 Average Linkage

Average Linkage defines the distance between two clusters as the average distance between the entries in each cluster

$$d_{C_i,C_j} = \operatorname*{mean}_{x,y} \{D(x,y) | x \in C_i,\ y \in C_j\}. \tag{2.42}$$

Average Linkage is the middle ground between Single Linkage and Complete Linkage. It represents a compromise between them and avoids the chaining property of Single Linkage, as well as the closeness property of Complete Linkage. Thus, it attempts to produce relatively compact clusters that are relatively far apart (Hastie et al., 2009).

2.6.2.4 Ward's Method

Ward's Method merges the pairs of clusters that lead to the minimum increase in total within-cluster variance after merging (Ward Jr, 1963). The distance is calculated as the increase in the squared error that would result from when the two clusters C_i and C_j are merged

$$d_{C_i,C_j} = \frac{m_i m_j}{m_i + m_j} \|c_i - c_j\|^2, \tag{2.43}$$

where the cluster sizes of C_i and C_j are defined as m_i and m_j, and where c_i, c_j are the centroids for each cluster. Ward's Method is biased towards creating larger clusters, but more robust to noise and outliers (Raffinot, 2018).

The different linkage methods can be depicted as a scale, illustrated in Figure 2.4, with extremes ends between naive and symmetrical $1/N$ clustering and highly asymmetrical Single Linkage

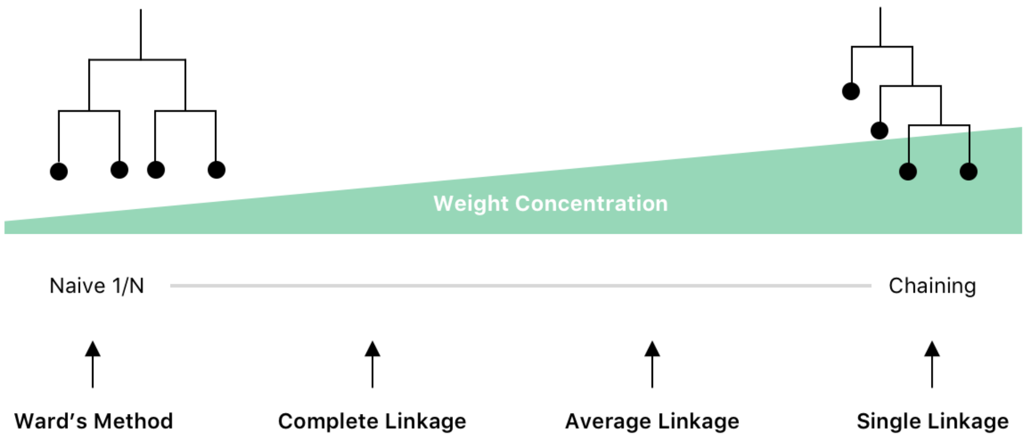

Figure 2.4: Symmetry properties of clusters derived using different linkage methods

(Papenbrock, 2011). In the empirical results found by Papenbrock (2011), Single Linkage and Ward's Method are deemed superior in terms of performance in the portfolio allocation methods explored. Of these methods, Single Linkage produces high concentration in weight allocation from the highly asymmetrical clusters but also results in a low correlation between clusters. Ward's Method is, on the other end, create high-quality, balanced clusters with low weight concentrations.

2.6.3 Optimal Number of Clusters

Clustering algorithms provide the actual clusters but do not handle the *optimal* number of clusters. As opposed to partitioning-based clustering methods, the hierarchical-based counterparts do not need to define the number of k clusters as an input, and instead find the cluster structure from 1 to N clusters in the N asset universe. However, this comes at a cost. Clusters are formed regardless of their reliability, considered a form of overfitting that potentially leads to spurious clusters being included (Raffinot, 2018).

For data segmentation problems such as correlation clustering, finding the optimal number of clusters k is a part of the problem. In this case, the cluster analysis is used to provide descriptive statistics for assigning an unknown number of natural clusterings, defined as k^*. Data-based methods for estimating k^* usually examine the within-cluster dissimilarity W_k as a function of the number of clusters $k \in \{1, 2, \ldots, k_{\max}\}$.

The Gap Statistic Index (GSI) method proposed by Tibshirani, Walther, and Hastie (2001) is commonly used to estimate k^*. It compares the logarithm of the empirical within-cluster dissimilarity $\log W_k$ with a corresponding curve obtained from uniformly distributed data with no apparent clusters. The within-cluster distance, W_k can be defined as the pooled within-cluster sum of squares around the cluster means

$$W_k = \sum_{r=1}^{k} \frac{1}{2n_r} D_r. \tag{2.44}$$

The sum of pairwise distances for all points in cluster r, D_r is defined as

$$D_r = \sum_{i,i' \in C_r} d_{ii'}. \tag{2.45}$$

The GSI measures the within-cluster dispersion around the cluster mean which is used to investigate the relationship between $\log W_k$ for different values of k compared to a suitable null reference distribution, $E_n^* \{\log W_k\}$. The difference between the distributions infers where the clustering provides the most information as compared to a distribution with no hierarchical structures or clusters. Thus, the gaps can be calculated as

$$\mathrm{Gap}_n(k) = E_n^* \{\log W_k\} - \log W_k. \tag{2.46}$$

Finally, to account for the dispersion of W_k as a function of the number of clusters, the optimization function is adjusted using the standard deviation to account for the varying volatility of estimates. More formally calculated as

$$s_k = \sqrt{(1 + 1/B)}\sigma(k), \tag{2.47}$$

where B represents the number of iterations of the reference distribution. The suggested number of clusters k by GSI can finally be inferred as

$$\begin{aligned} \min_k \quad & \mathrm{Gap}(k) \\ \text{s.t.} \quad & \mathrm{Gap}(k) \geq \mathrm{Gap}(k+1) + s_{k+1}. \end{aligned} \tag{2.48}$$

To apply this function, a large number of draws from the reference distribution needs to be computed to find reliable estimations of the distribution. There is a trade-off between computational speed and accuracy in this process. As the process of generating the reference distributions and running the clustering algorithm is computationally expensive, one has to weigh the reliability of the derived results with the computational costs. Yue, Wang, and Wei (2008) proposed an alternative function, bypassing the computationally expensive method described above by utilizing second-order differencing to find the optimal number of clusters, using the maximization function defined as

$$\begin{aligned} \max_{k} \quad & \{W_k - 2W_{k+1} + W_{k+2}\} \\ \text{s.t.} \quad & 1 \leq k \leq \sqrt{n}. \end{aligned} \tag{2.49}$$

The results are argued by the author to provide more stable results, less dependent on random draws, and less computationally expensive as there is no need to compute the reference distributions.

2.6.4 Optimal Leaf Ordering

For a given hierarchical tree with n leaves, there are 2^{n-1} linear orderings consistent with the structure of the tree. Hierarchical clustering algorithms usually employ heuristic methods of local similarities to generate the global ordering which does not necessarily lead to optimal leaf ordering (Bar-Joseph, Gifford, & Jaakkola, 2001). By using an optimal leaf ordering algorithm, the maximum sum of similarities of adjacent elements in the ordering can be achieved. This can be used to improve the suboptimal heuristic approach, illustrated in Figure 2.5.

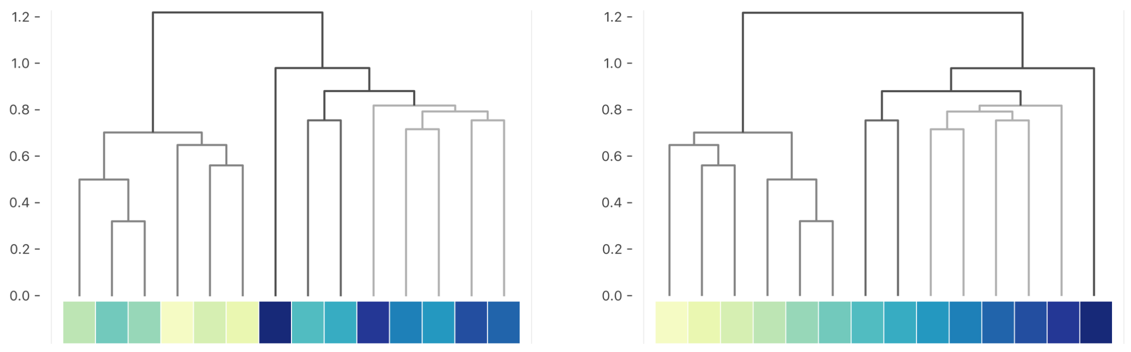

Figure 2.5: Dendrograms before optimal leaf ordering (left) and after optimal leaf ordering (right)

The objective of the algorithm employed, defined by Bar-Joseph et al. (2001), is to find the ordering of the tree leaves that maximizes the sum of similarities of adjacent leaves

$$D^{\phi}(T) = \sum_{i=1}^{n-1} \boldsymbol{S}(z_{\phi_i}, z_{\phi_{i+1}}) \tag{2.50}$$

where z_{ϕ_i} is the i^{th} leaf when T is ordered according to ϕ, and $\boldsymbol{S}$ is the similarity matrix. The optimization then find the ordering ϕ that maximizes $D^{\phi}(T)$. The increase in granularity of ordering of leaf-nodes increases the order of diagonalization in the covariance matrix, an essential feature in the later weight allocation step in the *Hierarchical Risk Parity* method presented in section 2.8.

2.7 Traditional Portfolio Allocation Methods

In this section, some of the traditional portfolio allocation methods are introduced, ranging from Markowitz's seminal work on Modern Portfolio Theory to the more modern methods of Maximum Diversification and Risk Parity.

2.7.1 Modern Portfolio Theory

Markowitz (1952) introduced the world to a new way to think about portfolio optimization with the introduction of the mean-variance framework. The most important part of his paper was the introduction of mean-variance efficient portfolios constructed as a convex optimization problem. This theory was expanded on with the publication of his subsequent book (Markowitz, 1959), where a more general model for portfolio selection, often referred to as Modern Portfolio Theory (MPT), was developed. Markowitz (1952) formulated the portfolio selection as a problem of finding a minimum variance portfolio of the assets in the investment universe that yields at least a target return R of expected return. Mathematically, this formulation can be expressed as the following *quadratic programming* problem

$$\begin{aligned} \min_{w} \quad & \boldsymbol{w}^\top \boldsymbol{\Sigma} \boldsymbol{w} \\ \text{s.t.} \quad & \boldsymbol{w}^\top \mathbf{1} = 1 \\ & \boldsymbol{w}^\top \boldsymbol{\mu} \geq R. \end{aligned} \tag{2.51}$$

where $\boldsymbol{w}$ is the weight of the assets in the portfolio and $\boldsymbol{\Sigma}$ is the covariance matrix of the assets. Since variance by definition is non-negative, it follows that the covariance matrix $\boldsymbol{\Sigma}$ is positive semidefinite, a critical criterion for Equation 2.51 to be a convex optimization problem (Kwan, 2010).

Markowitz (1952) approach requires the returns to be extremely accurate. While the methodology of how returns are estimated have become more sophisticated since the inception of MPT, it still poses a problem today. Braga (2015) highlighted that the approach has a poor out-of-sample performance because it favors investing in asset classes with high returns in comparison to the ones with low variances and negative correlations. Braga (2015) called this an *estimation error optimizer*. In general, portfolios constructed through the Markowitz model tend to produce concentrated weight distributions as the objective of the optimization is to optimize the diversification of volatility, not weight. This often leads to portfolios with much of the weight distributed to a relatively small selection of assets (Demey, Maillard, & Roncalli, 2010). This is a well-documented fact in the existing literature and is often attributed to estimation errors (Green & Hollifield, 1992). To alleviate some of the estimation errors, many practitioners have abandoned including expected returns altogether to instead focus on purely risk-based alternatives, solely using the covariance matrix as input.

2.7.1.1 Minimum-Variance Portfolio

Following the uncertain nature of return assumptions and the focus of risk-based portfolio optimization methods in this thesis, the global minimum-variance portfolio is further examined. This portfolio represents the portfolio on the efficient frontier that exhibits the lowest variance among all portfolios, disregarding any return constraint, effectively not including information about the expected returns at all (Munk, 2018). Thus, the portfolio can be defined by optimizing the below given optimization function

$$\begin{aligned} \min_{w} \quad & \boldsymbol{w}^\top \boldsymbol{\Sigma} \boldsymbol{w} \\ \text{s.t.} \quad & \boldsymbol{w}^\top \mathbf{1} = 1. \end{aligned} \tag{2.52}$$

Following the methodology above, this convex optimization problem can be solved analytically by restating it as a Lagrangian

$$\mathcal{L}[w, \lambda] = \frac{1}{2}\boldsymbol{w}^\top \boldsymbol{\Sigma} \boldsymbol{w} - \lambda(\boldsymbol{w}^\top \mathbf{1} - 1), \tag{2.53}$$

where the first-order conditions are

$$\frac{\partial \mathcal{L}[w, \lambda]}{\partial w} = \boldsymbol{\Sigma} \boldsymbol{w} - \lambda \mathbf{1} \tag{2.54}$$

$$\frac{\partial \mathcal{L}[w, \lambda]}{\partial \lambda} = \boldsymbol{w}^\top \mathbf{1} - 1. \tag{2.55}$$

Setting the first-order conditions to zero and solving for $\boldsymbol{w}$ and λ yields

$$\boldsymbol{w} = \lambda \boldsymbol{\Sigma}^{-1}\mathbf{1} \tag{2.56}$$

$$\lambda = \frac{1}{\mathbf{1}^\top \boldsymbol{\Sigma}^{-1}\mathbf{1}}. \tag{2.57}$$

The problem is finally solved by be solved by substituting in λ into $\boldsymbol{w}$ to get

$$\boldsymbol{w}_{MV} = \frac{\boldsymbol{\Sigma}^{-1}\mathbf{1}}{\mathbf{1}^\top \boldsymbol{\Sigma}^{-1}\mathbf{1}}. \tag{2.58}$$

This analytic solution is, however only available for the unconstrained, *global minimum-variance portfolio*. With additional constraints and bounds, the optimization problem is usually solved numerically. The global minimum variance portfolio, shown with its associated expected return, can be found in figure 2.6 depicted on the efficient frontier.

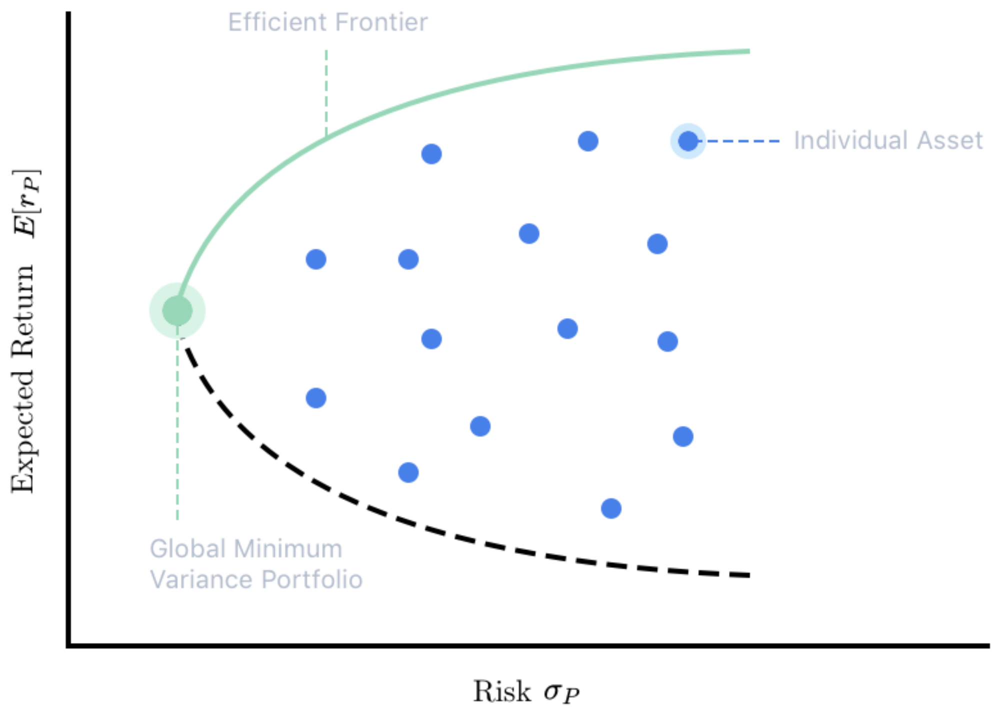

Figure 2.6: Global Minimum Variance Portfolio on the Efficient Frontier.

2.7.1.2 Long-Only Minimum-Variance Portfolio

In addition to the constraint of the weights summing to one, an additional constraint of long-only positions is applied to this study. Following the methodology of Munk (2018), the *long-only* *minimum-variance portfolio* is given by the solution to the following minimization problem

$$\begin{aligned} \min_{w} \quad & \boldsymbol{w}^\top \boldsymbol{\Sigma} \boldsymbol{w} \\ \text{s.t.} \quad & \boldsymbol{w}^\top \mathbf{1} = 1 \\ & w_i \geq 0 \end{aligned} \tag{2.59}$$

The optimization problem is thus to minimize the variance of the portfolio, constrained by all weights summing to one, and all weights having a minimum value of zero. This constrained optimization problem has no analytical solution, and therefore needs to be solved using numerical methods.

2.7.1.3 Inverse-Variance Portfolio

The inverse-variance portfolio is a special case of the minimum-variance portfolio, with the additional assumption that cross-correlations are all equal to zero, implying that the correlation and therefore also covariance matrix is diagonal. With these assumptions, the problem can be stated as

$$\begin{aligned} \min_{w} \quad & \boldsymbol{w}^\top \operatorname{diag}(\boldsymbol{\Sigma}) \boldsymbol{w} \\ \text{s.t.} \quad & \boldsymbol{w}^\top \mathbf{1} = 1. \end{aligned} \tag{2.60}$$

From the above optimization problem, the Lagrange method as applied in equation 2.53 can be used to arrive at

$$\boldsymbol{w}_{IV} = \frac{\operatorname{diag}(\boldsymbol{\Sigma})^{-1}\mathbf{1}}{\mathbf{1}^\top \operatorname{diag}(\boldsymbol{\Sigma})^{-1}\mathbf{1}}. \tag{2.61}$$

2.7.2 Risk Parity

While revolutionary for its time, Modern Portfolio Theory came with some inherent problems, namely, allocating a large portion of the weight to a relatively small subset of assets within the portfolio. This means that in case these few assets perform poorly during a period time, the entire portfolio will perform badly. Qian (2005) argues that risk is the true diversification factor, meaning that if a substantial portfolio loss can be attributed to a few assets, then the portfolio is not diversified. Furthermore, he instead suggests an alternative investment approach, namely Risk Parity.

Risk Parity (RP) is an investment approach that allocates volatility instead of capital. The method was first introduced to the retail investment market by Bridgewater in 1996, and the term was coined by Edward Qian in 2005. Qian (2005) argues that the investment portfolio should be risk-allocated instead of capital-allocated and that the traditional cross-asset heuristic approach (traditionally 60% equity, 40% bonds) is sub-optimal in terms of diversification. In addition, he argues that the risk contribution from stocks in the aforementioned portfolio is more significant than the risk contribution from bonds, implying that the potential majority loss would likely come from equities rather than bonds in a downturn. Ilmanen and Villalon (2012) contributes empirical grounds for this theoretical assumption and shows that the typical 60/40 equity/bonds portfolio has around 90% risk contribution attributed to equities.

By making sure that the loss contribution is the same for all components, the concept of risk contribution and its economic interpretation thus leads to the development of the *Equal Risk Contribution* (ERC) portfolio, which evenly allocates risk among different assets. Qian (2005) argues that if one subscribes to the philosophy that risk is indeed the true diversification factor, then the equal risk portfolio is the most diversified portfolio.

2.7.2.1 Equal Risk Contribution Portfolio

To understand the properties of the equal risk contribution (ERC) portfolios, Maillard et al. (2008) first define the marginal and total risk contributions of the different assets in the portfolio. They define the marginal risk contribution as the change in total risk if w_i increases by a small amount. Using this definition, the marginal risk contribution of asset i can be expressed as

$$MRC_i = \frac{\partial \sigma(\boldsymbol{w})}{\partial w_i} = \frac{(\boldsymbol{\Sigma w})_i}{2\boldsymbol{w}^\top \boldsymbol{\Sigma w}}, \tag{2.62}$$

where $(\boldsymbol{\Sigma w})_i$ is the i^{th} row of the vector from the product of $\boldsymbol{\Sigma}$ with $\boldsymbol{w}$ (Maillard et al., 2008). Using this, the total n marginal risk contributions in the vector can be defined as

$$MRC = \frac{\partial \boldsymbol{w}}{\sigma(\boldsymbol{w})}. \tag{2.63}$$

Intuitively, the total risk contribution of asset i can then be calculated by how much its risk is allocated into that asset, multiplied by its marginal risk contribution. The total risk contribution for asset i can thus be defined as the share of the total portfolio risk

$$RC_i = w_i \frac{(\boldsymbol{\Sigma w})}{\sigma(\boldsymbol{w})} = w_i MRC_i. \tag{2.64}$$

Maillard et al. (2008) asserts that the volatility is a homogeneous function of degree 1 and thus satisfies Euler's theorem and can be reduced to the sum of its arguments multiplied by their first partial derivatives. Following the methodology as presented by Maillard et al. (2008), the *total* *risk* of the portfolio can be defined as

$$TR = \sum RC_i = \boldsymbol{w}^\top \frac{\boldsymbol{\Sigma w}}{\sigma(\boldsymbol{w})} = \boldsymbol{\Sigma w}^\top \boldsymbol{\Sigma w} \tag{2.65}$$

Using the above definitions, the ERC portfolio can be constructed. As equal risk contribution implies that all assets should contribute an equal amount of risk to the total portfolio, the allocated capital is going to be determined by how risky the individual assets are. Following the methodology as presented by Maillard et al. (2008) for a long-only portfolio, the equal risk contribution portfolio problem must satisfy

$$\boldsymbol{w}_{ERC} = \Big\{ w_i \geq 0 : \boldsymbol{w}^\top \boldsymbol{1} = 1, w_i(\boldsymbol{\Sigma w})_i = w_j(\boldsymbol{\Sigma w})_j \quad \forall i, j = 1 \ldots n \Big\} \tag{2.66}$$

Maillard et al. (2008) emphasize that the above problem can be solved in different ways depending on assumptions of volatility and correlations. They consider three different scenarios for a portfolio of $n > 2$ assets;

1. Different volatilities but equal correlation
2. Different correlation but equal volatilities
3. Different volatilities and different correlations

In the first scenario, Maillard et al. (2008) assumes that the correlation between all assets are the same. The total risk contribution of component i can thus be written as

$$\sigma_i(\boldsymbol{w}) = \frac{w_i^2 \sigma_i^2 + \rho \sum_{j \neq i} w_i w_j \sigma_i \sigma_j}{\sigma(\boldsymbol{w})}. \tag{2.67}$$

The Equal Risk Portfolio is defined by $\sigma_i(\boldsymbol{w}) = \sigma_j(\boldsymbol{w})$ for all i, j. Using the fact that the constant correlation verifies that $\rho \geq -\frac{1}{n-1}$, Maillard et al. (2008) show that this is equivalent to $w_i \sigma_i = w_j \sigma_j$. Under these assumptions, the portfolio is the *inverse-volatility portfolio*, with the weights for asset i derived as

$$w_i = \frac{1/\sigma_i}{\sum_{j=1}^n 1/\sigma_j} \tag{2.68}$$

This is an intuitive solution as the weight allocated to an asset is inversely proportional to its volatility, allocating more weight to lower volatility assets as compared to high volatility assets. However, in the end, it is a naive solution that ignores any effects of potential cross-correlations.

In the next case, Maillard et al. (2008) assumes that the volatility of each asset is equal, but that their correlations differ, i.e., $\sigma_i = \sigma$ for all i. By employing the same process as in the previous case, they derive the weight of asset i as:

$$w_i = \frac{\left(\sum_{k=1}^{n} w_k \rho_{ik}\right)^{-1}}{\sum_{j=1}^{n} \left(\sum_{k=1}^{n} w_k \rho_{jk}\right)^{-1}} \tag{2.69}$$

Maillard et al., 2008 propose that because w_i is a function in and of itself, there is no clear solution to the above equation and thus no clear solution to the equal risk portfolio under the assumption that the volatility between assets is the same and correlations are different.

In the third and final case, Maillard et al. (2008) considers the case where both volatilities and correlations are different for each asset. The covariance between the return asset i and the return of the assets in the portfolio is given by

$$\sigma_{iw} = \text{cov}\left(r_{\text{i}}, \sum_j w_j r_j\right) = \sum_j w_j \sigma_{ij}. \tag{2.70}$$

The risk contribution of asset i can thus be expressed as

$$RC_i = \frac{w_i \sigma_{iw}}{\sigma(\boldsymbol{w})}. \tag{2.71}$$

Maillard et al. (2008) then introduces the beta β_i of asset i with the portfolio and defines it as $\beta_i = \frac{\sigma_i \boldsymbol{w}}{\sigma^2(\boldsymbol{w})}$. By combining this with the formula above, the follow expression for the risk contribution for asset i can be defined as

$$RC_i = w_i \beta_i \sigma(\boldsymbol{w}). \tag{2.72}$$

Applying the same thought process as in the previous cases, the authors finds that the weight of each asset in the equal risk portfolio is going to be defined by

$$w_i = \frac{\beta_i^{-1}}{\sum_{j=1}^{n} \beta_j^{-1}} = \frac{\beta_i^{-1}}{n}. \tag{2.73}$$

The above expression shows that the weight attributed to asset i is going to be inversely proportional to its beta. The higher the beta, the lower the weight, and vice versa. This implies that assets with high volatility or high correlation with other assets are penalized. Finding a solution for the above expression has no known analytical closed-form solution, and instead requires a numerical solution to solve the given optimization objective

$$\begin{aligned} \boldsymbol{w}_{ERC} &= \operatorname{argmin} f(\boldsymbol{w}) \\ \text{s.t.} \quad \boldsymbol{w}^\top \mathbf{1} &= 0 \\ w_i &\geq 0, \end{aligned} \tag{2.74}$$

where the objective function $f(\boldsymbol{w})$ is to minimize the variance of the risk contributions, given by the formula

$$f(\boldsymbol{w}) = \sum_{i=1}^{n} \sum_{j=1}^{n} \left(w_i (\boldsymbol{\Sigma w})_i - w_j (\boldsymbol{\Sigma w})_j \right)^2 . \tag{2.75}$$

Spinu (2013) proposes an alternative method to solve the problem, with similar solutions provided by Bai, Scheinberg, and Tutuncu (2016). In this case, the minimization problem can be reduced to an unconstrained quadratic optimization problem

$$\boldsymbol{x}^* = \operatorname*{argmin}_x \frac{1}{2} \boldsymbol{x}^\top \boldsymbol{\Sigma x} - \sum_{i=1}^{N} b_i \log(x_i), \tag{2.76}$$

where the marginal risk contributions are defined as

$$MRC_i(\boldsymbol{x}) \equiv b_i = 1/n. \tag{2.77}$$

The above solution optimize the function with regards to the vector $\boldsymbol{x}^*$. This vector can be considered the unbounded weights, and the final solution is derived by normalizing it between 0 and 1 through

$$\boldsymbol{w}_{ERC} = \frac{\boldsymbol{x}^*}{\sum_i^n x_i^*}. \tag{2.78}$$

Qian (2005) found empirical evidence for why one should allocate volatility instead of capital. While the risk-parity methodology can be applied to a large number of asset classes, the author chose to focus on how much capital should be invested into Equity/Bonds to demonstrate his point. Instead of investing 60/40 in Equity/Bonds, they found that by employing the RP methodology, they would instead invest 23/77 in Equity/Bonds. Table 2.1 further illustrates these findings.

	Russel 1000	**Lehman Agg**	**60|40**	**Parity**	**Parity (L)**
Average	8.3%	3.7%	6.4%	4.7%	8.4%
Standard Deviation	15.1%	4.6%	9.6%	5.4%	9.6%
Sharpe Ratio	0.55	0.80	0.67	0.87	0.87

Table 2.1: Returns Characteristics of Indices and Portfolios: 1983-2004 Source: Qian (2005)

As shown, the Sharpe ratio of investing according to capital was lower than the Sharpe ratio of individual bonds, which they attributed to poor diversification (as the overall portfolio's Sharpe is lower than one of its components). It can also be seen that the risk-parity portfolio has a higher Sharpe than both of the comparative portfolios. As previously discussed, the RP-portfolio is also trying to solve the problem of loss contribution to the portfolio and uses that as a measure of diversification. Qian (2005) found that the RP-portfolio's loss contribution was close to 50% (48/52 for stocks versus bonds), which shows that this goal has also been achieved.

2.7.3 Maximum Diversification

While Qian (2005) argued that the true measurement of diversification is risk, he did not offer any suggestion for how one could measure the diversification in said portfolio other than attributing risk equally to the assets in the portfolio. Choueifaty (2006) introduced the Maximum Diversification framework to academic literature and provided a ratio for how one could measure the diversification in the portfolio. Choueifaty and Coignard (2008) expanded on this framework by developing the maximum diversification (MD) portfolio. Choueifaty and Coignard (2008) propose that markets are risk-efficient in terms of *total risk* measured in volatility, in contrast to the CAPM assumption that only systematic, *non-diversifiable risk* is priced in. This method is based on the *diversification ratio* (DR), serving as an efficient alternative to the minimum-variance portfolio. Choueifaty, Froidure, and Reynier (2013) define the diversification ratio as the ratio of the portfolios weighted average volatility to its overall volatility

$$DR = \frac{\boldsymbol{w}^\top \boldsymbol{\sigma}}{\sqrt{\boldsymbol{w}^\top \boldsymbol{\Sigma} \boldsymbol{w}}}, \tag{2.79}$$

where $\boldsymbol{w}^\top$ is the vector of asset-weights and $\boldsymbol{\sigma}$ is the vector of asset volatilities,

$$\sum_{i=1}^{n} w_i \sigma_i = \boldsymbol{w}^\top \boldsymbol{\sigma}. \tag{2.80}$$

This measurement works as a proxy for diversification because, in a long-only portfolio, the volatility of the assets is less than or equal to the weighted sum of the assets' volatilities as the correlations are not perfectly positive. Thus, DR estimates the diversification that is gained by holding assets that are not perfectly correlated (Choueifaty et al., 2013). It makes intuitive sense that portfolios where the capital is spread out in a lot of uncorrelated assets also have a high DR. (Choueifaty et al., 2013) formalize this intuition by decomposing the DR of a portfolio into its *weighted-correlation* and *weighted-concentration* measures AC and CR

$$DR = \frac{1}{\sqrt{AC(1-CR)+CR}}. \tag{2.81}$$

AC is defined as the volatility weighted average correlation of the components in the portfolio

$$AC = \frac{\sum_{i\neq j}\left(w_i\sigma_i w_j\sigma_j\right)\rho_{i,j}}{\sum_{i\neq j}\left(w_i\sigma_i w_j\sigma_j\right)}, \tag{2.82}$$

and CR is the volatility weighted concentration ratio of the portfolio

$$CR = \frac{\sum_i \left(w_i\sigma_i\right)^2}{\left(\sum_i w_i\sigma_i\right)^2}. \tag{2.83}$$

A portfolio consisting of a single long position would have a CR of 1, while an equal volatility weighted portfolio has the lowest possible CR, equal to the inverse of the number of assets it contains. The DR of the portfolio therefore has an inverse relationship with AC and CR, and increases following the decrease in any of the two. The most diversified portfolio is thus be given by the case where the AC and CR are minimized.

2.7.3.1 Maximum Diversification Portfolio

Following the methodology of Choueifaty et al. (2013), the long-only maximum diversification portfolio (MDP) can be constructed by maximizing the following problem

$$\begin{aligned} \max_{\boldsymbol{w}} \quad & \frac{\boldsymbol{w}^\top \boldsymbol{\sigma}}{\sqrt{\boldsymbol{w}^\top \boldsymbol{\Sigma} \boldsymbol{w}}} \\ \text{s.t.} \quad & \boldsymbol{w}^\top \mathbf{1} = 1 \\ & w_i \geq 0 \end{aligned} \tag{2.84}$$

An alternative optimization approach can be utilized, more in line with the minimum-variance solution. Instead of maximizing the diversification ratio, the problem is instead stated as

$$\begin{aligned} \min_{x} \quad & \frac{1}{2}\boldsymbol{x}^\top \hat{\boldsymbol{\Sigma}} \boldsymbol{w} \\ \text{s.t.} \quad & \boldsymbol{x}^\top \hat{\boldsymbol{\sigma}} \ = 1 \\ & x_i \ \geq 0. \end{aligned} \tag{2.85}$$

The above optimization problem is thus stated as minimizing the variance, subject to the the weights being non-negative, and the weighted volatilities all summing to one. The unbounded weights represented by the $\boldsymbol{x}$ vector is normalized between 0 and 1 by dividing each x by the sum of the vector,

$$\boldsymbol{w}_{MP} = \frac{\boldsymbol{x}}{\sum_i^n x_i}. \tag{2.86}$$

Choueifaty et al. (2013) define the core properties of MDP:

1. Any stock not held by the MDP is *more* correlated to the MDP than any of the stocks that belong to it. Furthermore, all stocks belonging to the MDP have the same correlation to it.
2. The long-only MDP is the long-only portfolio such that the correlation between any other long-only portfolio and itself is greater than or equal to the ratio of their DR's.

Consider a portfolio that has been built using Russell 3,000 stocks. The MDP portfolio might hold 100 of these stocks, but it does not mean that the remaining 2,900 stocks have not been considered. The reason that the portfolio does not hold more (or less) assets is that the other stocks are more correlated to the MDP than the stocks it currently holds (Choueifaty et al., 2013).

2.7.4 Equally-Weighted Portfolio

The equally-weighted portfolio is the portfolio where all assets are given the same weight, that is, a weighted inverse to the number of assets in the portfolio

$$w_i = \frac{1}{n}. \tag{2.87}$$

The vector consisting of all the portfolio weights is defined as

$$\boldsymbol{w}_{EW} = (1/n, \ldots, 1/n)^\top. \tag{2.88}$$

Among others, DeMiguel et al. (2009) found substantial empirical grounds that despite its simplicity, the equally-weighted portfolio tends to outperform more sophisticated strategies regularly. More specifically, the authors find that of 14 models evaluated, the equally-weighted portfolio is consistently a top performer in terms of Sharpe ratio, certainty-equivalent return, and turnover in out-of-sample backtests. They propose that the gain made from optimal diversification is more than offset by its estimation error, rendering many methods consistently performing worse than this purely naive allocation method.

2.8 Hierarchical Risk Parity

Hierarchical Risk Parity (HRP) proposed by López de Prado (2016) utilizes developments in machine learning and graph theory to bypass the need for inversion of the covariance matrix, a leading issue in estimation of weights in portfolio optimization. The source of the problem can be attributed to the inherently unstable nature of matrix inversion of increasingly large covariance matrices as the condition number increases from the absolute difference from the two most extreme eigenvalues, as discussed in section 2.4. By relying on a more intricate algorithm, HRP manages to avoid inverting the covariance matrix and can create efficient allocations for ill-conditioned and even singular covariance matrices (López de Prado, 2016) following a three-step process:

1. Hierarchical clustering
2. Matrix seriation
3. Naive recursive bisection

One should note that the recursive bisection proposed by López de Prado (2016) does not incorporate any information about the hierarchies of the clustering results, only the ordering of the leaf-nodes themselves. Thus, the formation of the binary tree produced is determined by the number of assets and not the clusters or distances inferred by the hierarchical structure of the binary tree. Therefore, this recursive bisection is referred to as *naive recursive bisection.*

2.8.1 Hierarchical Clustering

The original method, defined by López de Prado (2016), uses the Single Linkage applied through the agglomerative hierarchical clustering algorithm. Using this linkage method yields the same results as the Minimum Spanning Tree (MST), a method derived from graph theory. The original approach proposed by López de Prado (2016) is enhanced in this paper using the *optimal leaf ordering* algorithm introduced in section 2.6 to create more robust clusters in the recursive bisection step.

2.8.2 Matrix Seriation

Matrix seriation is a statistical method used to reorganize rows or columns of a matrix to enumerate them in an appropriate order. Eisen, Spellman, Brown, and Botstein (1998) highlights how a data matrix can be reorganized efficiently using hierarchical clustering methods by enumerating the columns according to the order of the leaves. In this way, the seriation method sorts the correlation matrix using information obtained from the hierarchical clustering method.

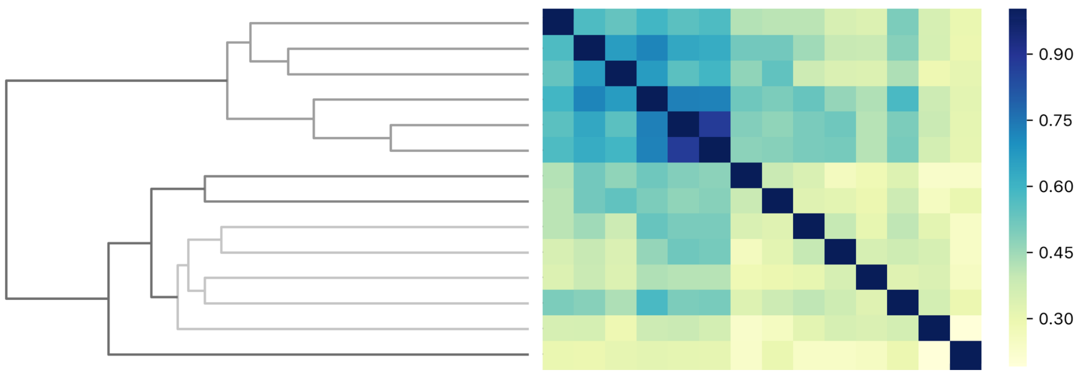

Figure 2.7: Illustration of the seriation of a covariance matrix using hierarchical clustering

López de Prado (2016) uses seriation as a way to rearrange the rows and columns of the covariance matrix into a more diagonalized representation by aligning the highest covariances along diagonal. In this way, similar investments are placed closer together and dissimilar investments further apart. The author refers to this structure as quasi-diagonal as higher covariance values are placed closer to the diagonal, and argues that the inverse-variance allocation is optimal in this case. Raffinot (2018) propose that this fact also can be expressed as the *Equal Risk Contribution* (ERC) defined in section 2.7.2 as it produces the same weight allocation while also relaxing the assumption of the cross-correlations being zero.

2.8.3 Naive Recursive Bisection

The recursive bisection algorithm uses information from the two previous steps to allocate weights to the assets in the portfolio. First, a list of items is initialized to represent the clusters

$$C = \{C_0\} \quad \text{with} \quad C_0 = \{n\}_{n=1,\ldots,N}. \tag{2.89}$$

Next, the between-cluster weights are initialized for each asset to 1

$$W_n = 1 \quad \forall n = \{\ 1, \ldots, N\}. \tag{2.90}$$

The naive recursive bisection method assumes a binary tree where clusters are recursively split into two *equally-sized* sub-clusters C_1 and C_2. Let $\mathbf{\Sigma}_i$ be the covariance matrix for cluster C_i.

The cluster variance can then be determined as

$$V_i = \boldsymbol{w}_i^\top \boldsymbol{\Sigma}_i \boldsymbol{w}_i, \tag{2.91}$$

where $\boldsymbol{w}_i$ is the vector of $w_n \ \forall n \in C_i$, and where the weight of each asset n is determined by the intra-cluster inverse-variance allocation

$$w_n = \frac{1/\sigma_n^2}{\sum_{j=1}^{N} 1/\sigma_j^2}. \tag{2.92}$$

The relative weight given to each of the sub-clusters C_1 and C_2 are determined by a split factor, α_1 and α_2, calculated using the inverse-variance allocation between the two clusters, which for two assets simplifies to their relative variances

$$\begin{aligned} \alpha_1 &= 1 - \frac{V_1}{V_1 + V_2} \\ \alpha_2 &= 1 - \alpha_1 \end{aligned} \tag{2.93}$$

where V_1 is the cluster variance of C_1, and V_2 is the cluster variance of C_2. The weights in the sub-clusters, defined as W_1 and W_2, are then updated according to the split factor

$$\begin{aligned} W_1 &:= \alpha_1 \times W_1 \\ W_2 &:= \alpha_2 \times W_2. \end{aligned} \tag{2.94}$$

The procedure is recursively executed from the top cluster that contains all assets until each asset is in its cluster. In this way, the weights are allocated in a top-down manner, where asset weights are based on both intra-cluster risk and inter-cluster risk.

In his original paper, it should be noted that the HRP portfolio is computed using the Single Linkage algorithm. Raffinot (2017) found that the HRP performance deteriorates with the use of other linkage methods.

2.9 Hierarchical Equal-Risk Contribution

Hierarchical Equal-Risk Contribution (HERC) is a clustering based portfolio allocation method introduced by Raffinot (2018) aimed to build onto his previous work on the Hierarchical-Based Asset Allocation (Raffinot, 2017) by augmenting it with the recursive bisection approach presented by López de Prado (2016). It follows a similar structure to the HRP method and uses the following steps:

1. Hierarchical correlation clustering
2. Estimation of optimal number of clusters
3. Hierarchical recursive bisection-based weight allocation between clusters
4. Naive risk parity weight allocation within clusters

In addition to variance-based risk metrics for allocation, Raffinot (2018) extends the HERC methodology to cover *conditional value at risk* (CVaR) and *conditional drawdown at risk* (CDaR). However, this study only focus on the case with variance-based risk.

The main differentiator between HRP and HERC is how the recursive bisection is carried out. HRP assumes a naive approach where the dendrogram produced by the clustering step is ignored, and only the structure of the leaf-nodes is taken into account. HERC extends the naive approach found in HRP by splitting the bisections according to the dendrogram, thus avoiding splitting natural clusters up.

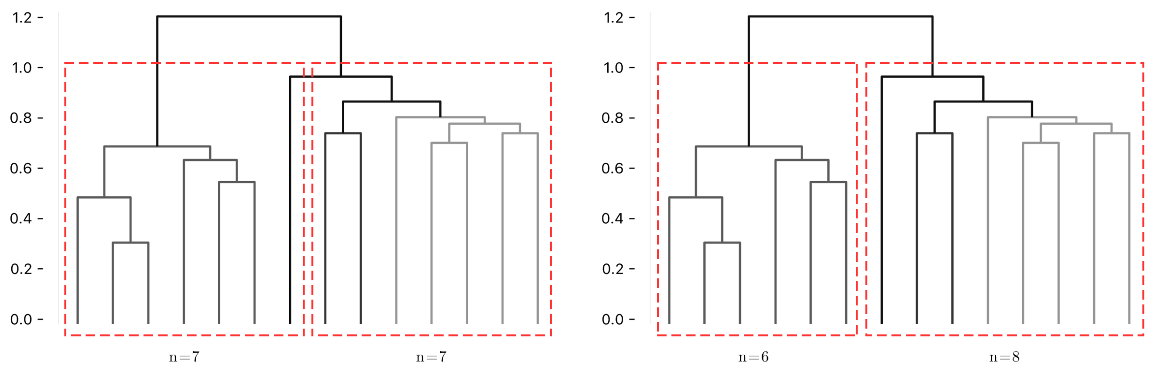

Figure 2.8: Assets divided into 2 clusters implied by naive recursive bisection on the left, and clusters implied by hierarchical recursive bisection on the right

Raffinot (2018) found that the HERC portfolio provides statistically better risk-adjusted performances than common portfolio optimization techniques. In contrast to the HRP portfolio, Raffinot (2018) found that Ward's Method yields the best results for HERC. Below, the four steps are examined in further detail.

2.9.1 Hierarchical Clustering

The hierarchical clustering follows the theory outlined in 2.6. The full tree graph is created using the agglomerative hierarchical clustering algorithm which, is then used to sort the covariance matrix (equivalent to matrix seriation).

2.9.2 Optimal Number of Clusters

The estimated optimal number of clusters is derived from the Gap Statistic (Raffinot, 2018), previously examined in section 2.6. The optimal number of clusters determines how deep the recursive bisection goes in the subsequent recursive bisection step.

2.9.3 Hierarchical Recursive Bisection

The information retrieved from all previous steps is used to allocate weights to the assets in the portfolio using the recursive bisection. The method is similar to that of HRP, but use the dendrogram structure to infer the cluster structure instead of a equally-sized approach. First, a list of items is initialized to represent the clusters

$$C = \{C_0\} \quad \text{with} \quad C_0 = \{n\}_{n=1,\dots,N} \tag{2.95}$$

Next, the between-cluster weights are initialized for each asset to 1

$$W_n = 1 \quad \forall n = \{\ 1, \dots, N\}. \tag{2.96}$$

As previously stated, the hierarchical recursive bisection uses the dendrogram derived from the hierarchical clustering step to bisect each cluster into sub-clusters C_1 and C_2 until the optimal number of clusters k is reached. The cluster weights are calculated according to the ERC allocation

$$\alpha_1 = \frac{RC_1}{RC_1 + RC_2} \quad , \quad \alpha_2 = 1 - \alpha_1. \tag{2.97}$$

RC_i represents the risk contribution of each cluster, defined as the cluster variance

$$RC_i = \boldsymbol{w}_i^\top \boldsymbol{\Sigma}_i \boldsymbol{w}_i, \tag{2.98}$$

where $\boldsymbol{w}_i$ is the vector of $w_n\ \forall n \in C_i$. The weight of each asset n is determined by the intra-cluster inverse-variance allocation

$$w_n = \frac{1/\sigma_n^2}{\sum_{j=1}^{N} 1/\sigma_j^2}. \tag{2.99}$$

The weights W_1 and W_2 in the sub-clusters C_1 and C_2 are then updated according to the split factor.

$$\begin{aligned} W_1 &:= \alpha_1 \times W_1 \\ W_2 &:= \alpha_2 \times W_2 \end{aligned} \tag{2.100}$$

This procedure is recursively executed from the top cluster that contains all assets until it reaches the estimated optimal number of clusters k^* determined by the Gap statistic.

2.9.4 Within-Cluster Weight Allocation

In the last step, the between-cluster weights are multiplied with the within-cluster weights to derive the weight of each asset. First, define the within-cluster weights $\boldsymbol{w}_i$ of each cluster $C_i \in [1, \ldots, k^*]$, where $\boldsymbol{w}_i$ is the vector of $w_n \ \forall n \in C_i$, and where the weight of each asset n is determined by the intra-cluster inverse-variance allocation

$$w_n = \frac{1/\sigma_n^2}{\sum_{j=1}^{N} 1/\sigma_j^2}. \tag{2.101}$$

Let $\boldsymbol{W}$ be the vector of all between-cluster weights and $\boldsymbol{w}$ be the vector of all within-cluster weights, then the weight for each asset can be determined as

$$\boldsymbol{w}_{HERC} = \boldsymbol{W} \cdot \boldsymbol{w} \tag{2.102}$$

2.10 Nested Clustered Optimization

Nested Clustered Optimization (NCO) introduced by López de Prado (2019) is a machine learning-based approach to tackle the structural problems of covariance instability in modern portfolio theory. The end-goal of the method is to reduce the ill-conditioned optimization problem into several smaller and more well-behaved sub-problems. The algorithm can be broken down into four steps:

1. Correlation clustering
2. Estimation of optimal number of clusters
3. Intra-cluster weight allocation
4. Inter-cluster weight allocation

2.10.1 Correlation Clustering

López de Prado (2019a) utilizes a K-means clustering algorithm for correlation clustering. For the sake of comparability between NCO, HRP, and HERC, the agglomerative hierarchical clustering method is instead applied in this thesis, following the theory outlined in 2.6.

2.10.2 Optimal Number of Cluster

The results from the hierarchical clustering step are used in conjunction with the optimal number of clusters to split up the covariance matrix into several cluster-based covariance matrices. The optimal number of clusters are estimated using the method outlined in section 2.6.3.

2.10.3 Within-Cluster Weight Allocation

Each cluster-based covariance matrix derived from the previous step is separately allocated within-cluster weights according to the applied portfolio optimization method. In the author's original example, the unconstrained minimum-variance allocation is used

$$\boldsymbol{w}_i = \frac{\boldsymbol{\Sigma}_i^{-1}\mathbf{1}}{\mathbf{1}^\top \boldsymbol{\Sigma}_i^{-1}\mathbf{1}}, \tag{2.103}$$

where $\boldsymbol{\Sigma_i}$ represents the the covariance matrix for cluster i. However, López de Prado (2019a) states that the allocation method is flexible to the actual underlying allocation method, and nothing prevents the use of any alternative method outlined in section 2.7, or other portfolio optimization methods.

2.10.4 Between-cluster Weight Allocation

The third and final step is to compute the inter-cluster weights. Let $\boldsymbol{\Sigma}_i$ be the covariance matrix for cluster C_i. The cluster variance V_i can then be determined as

$$V_i = \boldsymbol{w}_i^\top \boldsymbol{\Sigma}_i \boldsymbol{w}_i \tag{2.104}$$

The within-cluster portfolio variance of each cluster is then used to construct the reduced covariance matrix

$$\boldsymbol{\Sigma} = \begin{pmatrix} V_1 & v_{12} & \dots & v_{1n} \\ v_{21} & \ddots & \dots & \vdots \\ \vdots & \vdots & \ddots & \vdots \\ v_{n1} & \dots & \dots & V_n \end{pmatrix} \tag{2.105}$$

where n is the number of clusters and v_{ij} is the covariance between cluster i and j. The reduced covariance matrix is used to compute the minimum-variance allocation from equation 2.103 between the clusters

$$\boldsymbol{W} = \frac{\boldsymbol{\Sigma}^{-1}\mathbf{1}}{\mathbf{1}^{\top}\boldsymbol{\Sigma}^{-1}\mathbf{1}}. \tag{2.106}$$

Let $\boldsymbol{w}$ be the combined vector of within-cluster weights for all assets determined in the previous step. The within-cluster weights and the between-cluster weights can then be used to compute the asset weights as

$$\boldsymbol{w}_{NCO} = \boldsymbol{W} \cdot \boldsymbol{w}. \tag{2.107}$$

López de Prado (2019a) argues that since similar assets are reduced to clusters, the reduced covariance matrix is by construction closer to a diagonal matrix, and thus closer to the optimal solution of the original convex optimization proposed by Markowitz (1952).

3 Data

In this section, the applied investment universe is presented, followed the data collection and data processing procedures to acquire and clean the data. Secondly, the risk-free rate is discussed. Third and Finally, the synthetic data generation methodology for the Monte Carlo method is presented.

3.1 Sample Selection

The data selection in this study is based on a sample of 4,677,831 observations of daily pricing data on U.S. equities featured in the S&P 500 index between the period 1990-01-02 to 2020-01-17. Once included in the index, the constituents are kept as available assets throughout the timeline as long as they fulfill the other criteria, described further in section 3.2.

The timeframe was chosen to balance the trade-off being representative of future performance and to cover different macroeconomic regimes and tail events. The future market expectations compared to historical returns are found to have higher correlations as the historical returns are closer to the present time, which warrants a more narrow timeframe to be utilized (Greenwood & Shleifer, 2014). However, the timeframe between 1990 and 2020 includes several significant macroeconomic events, including but not limited to the Tech-Bubble around 2000 and the 2008 Financial Crisis, increasing the validation of empirical results over a wider range of market regimes.

3.1.1 Investment Universe

All historical constituents of the S&P 500 between the beginning of January 1990 to the mid January 2020 have been included. The main reason to use U.S. equity in general, and the S&P 500 constituents, in particular, is that this market represents the largest and most liquid capital market in the world. Figure 3.1 illustrate this point by contrasting the relative size of the U.S. equity market with the eight other largest equity markets around the world.

To research how capital can be allocated efficiently in a more robust and higher performing fashion, the liquidity and size of the market create a solid foundation upon which empirical findings can be well-grounded. Another key reason for restricting the study to the U.S. equity

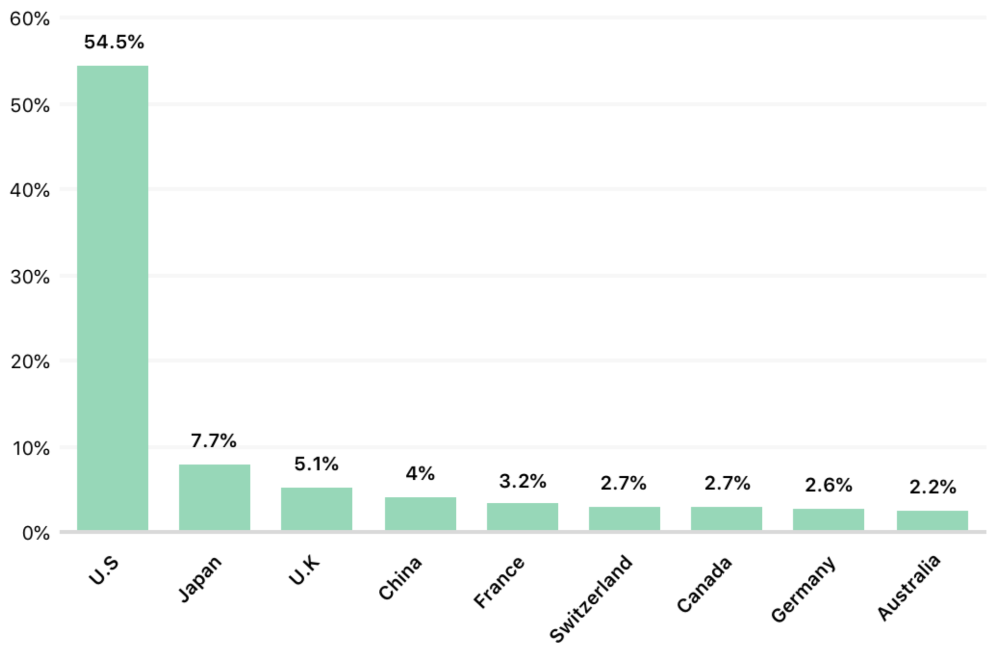

Figure 3.1: U.S market share of world equity capital markets

Source: Credit Suisse (2020)

market is that it eliminates currency risks and reduces complexity in data-handling of currency exchange effects.

All historical constituents in the S&P 500 index from January 2nd 1990 until January 17th 2020 are included to mitigate potential survivorship-bias and look-ahead bias. Survivorship bias is defined as the logical fallacy of concentrating on the stocks that still exist today to predict future performance. If only current constituents were to be included, the investment universe would be overrepresented by 'winners' as opposed to 'losers'.

A minimum data requirement of 24 months of data has been applied to take the correlation and covariance estimation in the portfolio allocation method into account. In addition, for an asset to be included in the rebalancing, 24 prior months need to be available for the same reason.

3.1.2 S&P 500 General Information

S&P 500 includes the largest 500 listed companies in the U.S. equity market. The index is capitalization-weighted, meaning that its market capitalization determines the weight of each asset as a fraction of the total market capitalization of all 500 companies. The S&P 500 covers approximately 80 percent of the total market capitalization of the U.S. stock market, thus representing the overall U.S. equity market reasonably well.

There are several versions of the S&P 500 index; *price return* as well as the *total return* index. The difference between these two is that the total return index includes total reinvestment of any cash distributions from the assets. For this thesis, the equally-weighted total return index is used to assess the actual performance of holding the assets, included as a benchmark portfolio in the empirical results. To be included in this index, the stocks have to fulfill several criteria that are included in Appendix A1.

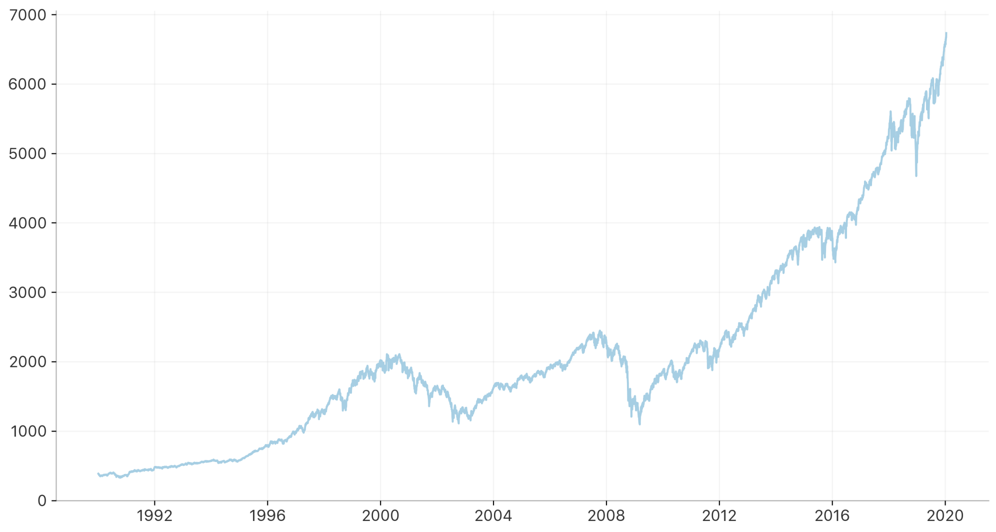

Figure 3.2: Historical Price Fluctuations of the SP 500 Capitalization-Weighted Total Return Index

Source: Thomson Reuters Eikon (2020)

3.2 Data Collection and Data Processing

The price and return data used were obtained from Compustat North America Security Daily and cover the asset universe of the total constituents in the index from January 2nd, 1990 to January 17th, 2020. The gross investment universe was filtered in several steps to get the net list of assets used. First, companies that changed ticker over the investment horizon was merged into a single issue. Secondly, companies with several issues of common stocks were reduced to a single stock, with a heuristic selection process primarily based on the availability of pricing data of the issue (number of days with price data) and secondary on the liquidity of the issue based on trading volume. Thirdly, companies with less than 24 months of available price data were removed. Finally, missing price data was forward filled from the previous most recent day of available price data for the same stock.

From the original gross universe of 1620 assets, 688 issues of duplicate listings were removed. In addition, 3 listings representing the same asset were merged. Finally, 26 listings with less than 24 months of price data were removed. This results in a net universe of 903 assets. Also, the price data is adjusted for corporate actions that affect the market price but not the value of the investment return such as dividends, share repurchases, spinoffs, and stock splits. The daily adjusted closing price is thus calculated taking the actual closing price and adding back any effect from above said actions.

3.3 Risk-Free Rate

There is no consensus regarding the choice of the risk-free rate, and the different frequently used rates for the U.S. market include the 10Y-government bond, 5Y-government bond, 2Y-government bond, 3-month LIBOR, 1-month LIBOR and the Overnight Index Swap (OIS).

The reason that the 10Y-government bond, 5Y-government bond, or 2Y-government bond are not used as a proxy for the risk-free rate is because of their time uncertainty. A rate that is 10, 5, or 2 years down the line is riskier than it would be for the other rates that were mentioned. This can also be seen by looking at the yield levels for the different interest rates. It is important to mention that the 10-year yield level is not always higher than the other mentioned bonds. One metric that is heavily quoted in the media, is that during uncertain times, the 10-year government bond yield is lower than the 2-year government bond yield, which implies that investors feel like the short term risk (2 years) is higher than the long term risk (10 years) and thus require higher compensation. This is also true for the other rates that are mentioned, during times of crisis, the dynamics for the different rates changes. For example, Duffie and Stein (2015) showed that during the financial crisis in 2008, the spread between the 3-month Overnight Index Swap and 3-month LIBOR widened.

The 1-month LIBOR is used as a proxy for the risk-free rate in this paper. LIBOR is short for London Interbank Offered Rate and is an unsecured short-term borrowing rate between banks. The maturity times range from one single day to a year and serve as a reference rate for transactions throughout the world. LIBOR rates are compiled by asking 18 global banks to provide quotes estimating the rate of interest at which they could borrow funds from other banks at 11:00 A.M. UK time. The highest four and lowest four values are discarded, and the remaining ones are averaged to determine the rate.

3.4 Synthetic Data Generation

In addition to the historical price data from the S&P 500 constituents, additional data are synthetically created for the Monte Carlo method experiments for robustness tests and theoretical asymptotic results. The created returns are then fit to follow a multivariate Gaussian distribution with covariance matrices created in two different ways. First, block-diagonal correlation matrices are constructed that represents the theoretically *true* correlation matrix, which is used to measure the difference in weight allocation between the true and the empirically observed correlation matrix. Secondly, correlation matrices are generated through a Deep Learning algorithm, trained using observed correlation structures in the S&P 500 constituent returns to form realistic but not historically bound correlation structures. The correlation matrices are then used to infer a covariance matrix used by the portfolio optimization methods.

3.4.1 Block-Diagonal Correlation Matrix

The block-diagonal correlation matrix represents a stylized representation of the true correlation matrix of the S&P 500, highlighting the hierarchical correlation structure within clusters. This is achieved by creating one or several clusters of assets that have a non-zero correlation within the cluster, and zero correlation outside the cluster, and where each block is placed along the diagonal of the matrix.

In addition to the originally proposed correlation matrix an additional more realistic block-diagonal correlation matrix is applied. Initially, a random number of clusters between 5 and 10 is produced of varying sizes that sums to 100 assets. Furthermore, the clusters are placed along the diagonal to form the correlation matrix with non-zero between-cluster correlations. An example correlation matrix with 5 clusters is illustrated on the right-hand-side in Figure 3.3.

From the above described correlation matrices, a *true* covariance matrix can be inferred using the methodology from López de Prado (2020). First, the columns are randomly shuffled to hide the

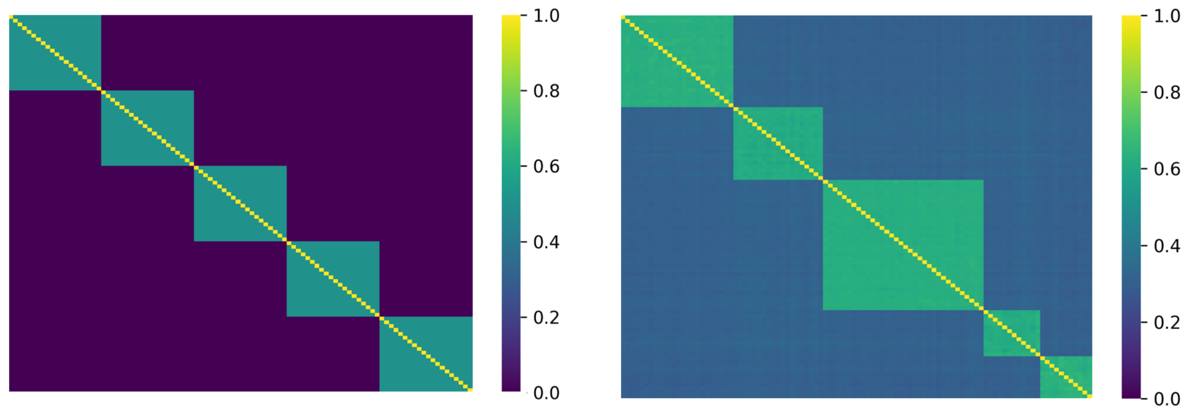

Figure 3.3: Heatmap of a Block-Diagonal Correlation Matrix of 100 assets

cluster structure. Then, the variance applied to the covariance matrix is drawn from a lognormal distribution fitted to the distribution of volatility of the S&P 500 return data described in section 3.1. Using this method, two highly stylized examples of a financial covariance matrix can be constructed and used to infer the estimation error in empirical tests.

3.4.2 CorrGAN

Marti (2019) introduced a novel approach to solve the problem of generating realistic financial correlation matrices, referred to as CorrGAN. The method applies a Generative Adversarial Network (GAN) to generate artificial but realistic correlation matrices by learning the implicit general structure of the training sample, which in this case, is the empirical correlation matrices. To better understand the process, the general methodology of GAN is first examined.

In a Generative Adversarial Network, proposed by Goodfellow et al. (2014), two models are trained simultaneously by an adversarial process. A generative model G learns to capture the data distribution, and a discriminative model D learns to tell the real data distribution apart from the one generated by G. During training, G becomes incrementally better at replicating the distributions from the training data, while D become better at telling them apart. This continues until an equilibrium is reached where D can no longer distinguish the real empirical distributions in the training data from the synthetically created ones.

The empirical correlation matrices used as training data are generated from the S&P 500 returns sorted using the hierarchical clustering and matrix seriation techniques explained in section 2.7. The sampling window is reduced from the full timeline to the last 6 years of return data. This limitation of the timeframe used was made to better reflect the hierarchical structure and the stylized facts of correlation matrices by today's standard. The sample correlation matrices were generated using 504 days (approximately 2 years) of return data taken from a random starting time within the sampling window. Within that window, assets with missing return data are removed, and from that subset, 100 assets are randomly sampled.

From the training data, the CorrGAN algorithm is trained over 13,000 iterations to converge towards the realistic samples collected. The generated matrices from the model are, however, not correlation matrices as their diagonal is not precisely equal to one, and the matrices are not perfectly symmetric. The generated matrices are therefore post-processed using an alternating projection method to find the "nearest" theoretically correct correlation matrix (Marti, 2019). This is achieved using the method formalized by Higham (2002) using the notion of nearness from a weighted Frobenius norm to compute the nearest correlation matrix.

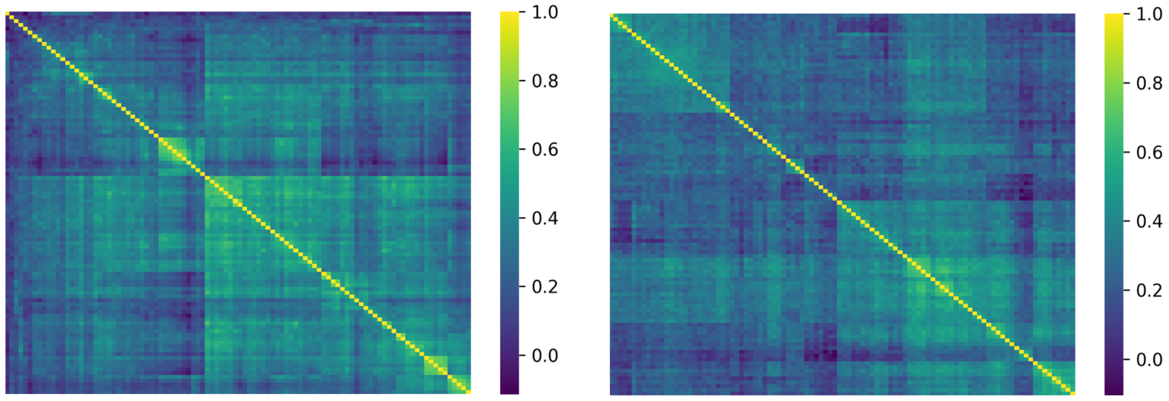

Figure 3.4: Comparison of heatmap of empirically observed correlation matrix to the left and synthetically created correlation matrix to the right.

Figure 3.4 depicts the side-by-side comparison between an empirical correlation matrix from the training data (left) and a synthetic correlation matrix generated from the CorrGAN method (right). From the correlation matrix derived through the CorrGAN method, a covariance matrix is inferred using the same methodology as described in section 3.4.1, where the columns are randomly shuffled and a standard deviation is applied from a lognormal distribution fitted to the observed S&P 500 volatility of returns.

4 Method

This chapter presents the research approach applied in the thesis Furthermore, the practical implementation of the theoretical frameworks is described. Finally, the implementation of the Monte Carlo method and the historical backtesting methodology is illustrated.

4.1 Research Method

4.1.1 Research Approach

A quantitative method was chosen to best investigate the problem of the thesis. The quantitative research approach emphasize on the quantification the collection and analysis of data, subsequently used to testing formalized theories and hypotheses (Bell, Bryman, & Harley, 2018). Furthermore, a deductive approach was applied as a clear hypothesis and theoretical framework could be formalized, and subsequently tested using gathered empirical data. From this, the research approach can be broken down into three steps. First, the previous literature was studied. Secondly, the initial reading of existing theories, an initial hypothesis regarding the result of the study was developed. Third and finally, the proposed hypothesis was tested following the methodology presented in this section.

4.1.2 Research Perspective

The paper is aims to replicate the behavior and focus of an institutional investor. Investment institutions can be entities such as pension funds, banks, among others, that provide financial services and advice to individuals through a pooled investment vehicle. In practice, several investment restrictions are implemented to better reflect the nature of the practical applicability of the investment strategies. All assumptions and restrictions regarding data is presented in section 3, and the practical implementation of the investment methods are presented further in the methodology.

4.1.3 Research Model

The research is conducted with an emphasis on the possible benefits of the hierarchical clustering in portfolio optimization, benchmarked, and contrasted to more traditional non-hierarchical methods. The empirical testing and evaluation of the different methods are conducted using Monte Carlo methods together with a walk-forward historical backtest.

4.2 Calculation Methodology

In this section, the methodology of calculating returns, volatility, and the covariance matrix are presented. All calculations are implemented using the Numpy and Pandas packages in Python.

The estimated return over any given period is calculated using the multiplicative geometric return, defined as

$$r_{t,n} = (1 + r_{t+1})(1 + r_{t+2}) \dots (1 + r_{t+n}) - 1, \tag{4.1}$$

where r is the simple rate of return $r_t = P_t/P_{t-1} - 1$, and n is the number periods observed. From this, the average return is defined as

$$\mu_i = \left(\prod_{t=1}^{T} (1 + r_{it}) \right)^{\frac{1}{T}} - 1 \tag{4.2}$$

where T defines the number of periods. As the observed returns are measured on a daily basis, this metric would indicate the daily geometric average return. However, it can also be stated in annualized form as

$$\mu_i = \left(\prod_{t=1}^{T} (1 + r_{it}) \right)^{\frac{252}{T}} - 1, \tag{4.3}$$

where 252 represents all trading days over a year calendar year. When calculating the variance and standard deviations, the standard implementation in Python has been implemented. Using this, the formula provides the maximum likelihood estimate of the variance for normally distributed variables using the following formula

$$\sigma_i^2 = \operatorname{Var}(r_i) = \frac{1}{T} \sum_{t=1}^{T} (r_{it} - \bar{r}_i)^2 \tag{4.4}$$

From this, the standard deviation can be derived, representing the statistical measure of asset volatility, which measures how widely the prices are deviating from the average price of the asset. As with variance, the maximum likelihood estimation is applied to the standard deviation calculation, and can thus be defined as

$$\sigma_i = \mathrm{Std}(r_i) = \sqrt{\sigma_i^2} \tag{4.5}$$

The covariance matrix is implemented using the following definition

$$\rho_{ij} = \mathrm{cov}(r_i, r_j) = \frac{1}{T}\sum_{t=1}^{T}(r_i - \bar{r}_i)(r_j - \bar{r}_j). \tag{4.6}$$

The correlation matrix follows the Pearson product-moment correlation coefficients and can be expressed as the normalized and demeaned covariance matrix

$$\rho_{ij} = \mathrm{corr}(r_i, r_j) = \frac{\sigma_{ij}}{\sigma_i \sigma_j}. \tag{4.7}$$

The shrinkage estimation applied to the covariance matrix follows the theory by Ledoit and Wolf (2004) and is implemented in Python using the PyPortfolioOpt package using the constant correlation model assumption.

4.3 Hierarchical Clustering

The agglomerative hierarchical clustering algorithm used for the hierarchical clustering-based portfolio optimization methods is implemented using the Scipy package in Python, following the structure outlined by Müllner (2011). This creates all necessary calculations and output in terms of creating the binary tree and the optimal leaf ordering. However, before computing the clusters, the sample correlation matrix has to be converted to Euclidean distances

$$d_{ij} = \sqrt{\frac{1}{2}(1 - \rho_{ij})}. \tag{4.8}$$

The distance matrix is then applied to get the pairwise distances, defined as

$$\tilde{d}_{ij} = \tilde{d}[D_i, D_j] = \sqrt{\sum_{n=1}^{N}(d_{ni} - d_{nj})^2}. \tag{4.9}$$

The pairwise distances are subsequently used as input to the hierarchical clustering algorithm. The linkage methods used in the study are restricted to three of the four methods outlined in section 2.6, as they are closest to the original implementations:

- Single Linkage
- Average Linkage
- Ward's Method

The output of the algorithm is a binary tree that covers all clustering levels from one cluster covering all N assets, to N singleton clusters, each containing one single asset. However, this method does not include the estimation of the optimal number of clusters, and the implementation of this method is therefore described in the following section.

4.3.1 Optimal Number of Clusters

Estimation of the optimal number of clusters k^* for the hierarchical clustering is computed using the theory outlined in section 2.6.3. Using this method, the within-cluster distances of the empirically observed data are computed according to

$$W_k = \sum_{r=1}^{k} \frac{1}{2n_r} D_r,$$

where the sum of pairwise distances for all points in cluster r, D_r is defined as

$$D_r = \sum_{i,i' \in C_r} d_{ii'}. \tag{4.10}$$

The pooled within-cluster sum of squares W_k is then subsequently used to compute the optimal number of clusters. As the original implementation of the Gap statistic is computationally expensive, the alternative method provided by Yue et al. (2008) is used, using the maximizing function

$$\begin{aligned} \max_{k} \quad & \{W_k - 2W_{k+1} + W_{k+2}\} \\ \text{s.t.} \quad & 1 \leq k \leq \sqrt{n}. \end{aligned} \tag{4.11}$$

The above method takes the second-order difference to find where the largest gain can be achieved. This method is used over the original GSI, as the computational cost is considerably lower, and therefore allows for more robust testing given the timeframe of the study.

4.4 Traditional Portfolio Allocation Methods

The traditional long-only portfolio allocation methods examined in this study are all solved using a numerical method (with the exception of the equally weighted and inverse-variance portfolios). All of the objective functions can be stated as a convex function of the weights given the supplied covariance matrix is positive semi-definite, and can thus be solved using convex optimization techniques through the Cvxpy package in Python. The full implementation of all portfolios can be found in Appendix A3.

4.4.1 Minimum-Variance Portfolio

The long-only constrained minimum variance portfolio is derived using the aforementioned method using the following objective function

$$\begin{aligned} \min_{w} \quad & \frac{1}{2}\boldsymbol{w}^\top \hat{\boldsymbol{\Sigma}} \boldsymbol{w} \\ \text{s.t.} \quad & \boldsymbol{w}^\top \mathbf{1} = 1 \\ & w_i \geq 0 \end{aligned} \tag{4.12}$$

The above expression can be summarized to minimize the expected variance of the portfolio, given the constraints that the weights have to sum to one and that all weights have to be non-negative.

4.4.2 Equal Risk Contribution Portfolio

For the long-only equal risk contribution portfolio, the original constrained minimization problem can be reduced to an unconstrained quadratic optimization problem

$$\min_{x} \quad \frac{1}{2}\boldsymbol{x}^\top \hat{\boldsymbol{\Sigma}} \boldsymbol{x} - \sum_{i=1}^{N} b_i \log(x_i), \tag{4.13}$$

where the marginal risk contributions are defined as

$$MRC_i(x) \equiv b_i = 1/n. \tag{4.14}$$

To find the final weights, the unbounded $\boldsymbol{x}$ vector is normalized between 0 and 1 by dividing each x^* by the sum of the vector, defined as

$$\boldsymbol{w}_{ERC} = \frac{\boldsymbol{x}}{\sum_i^n x_i}. \tag{4.15}$$

The above expression can be summarized as finding the long-only portfolio with minimum variance that satisfies all assets contributing to the same risk, all weights being positive, and finally, all weights summing to one.

4.4.3 Maximum Diversification Portfolio

The objective of the maximum diversification portfolio (MD) is to maximize the diversification ratio, DR, using an optimization defined as

$$\begin{aligned} \max_{\boldsymbol{w}} \quad & \frac{\boldsymbol{w}^\top \hat{\boldsymbol{\sigma}}}{\sqrt{\boldsymbol{w}^\top \hat{\boldsymbol{\Sigma}} \boldsymbol{w}}} \\ \text{s.t.} \quad & \boldsymbol{w}^\top \mathbf{1} = 1 \\ & w_i \geq 0 \end{aligned} \tag{4.16}$$

To ensure that the problem is solvable using convex optimization, the above maximization problem is restated as the below given minimization problem.

$$\begin{aligned} \min_{x} \quad & \frac{1}{2} \boldsymbol{x}^\top \hat{\boldsymbol{\Sigma}} \boldsymbol{w} \\ \text{s.t.} \quad & \boldsymbol{x}^\top \hat{\boldsymbol{\sigma}} = 1 \\ & x_i \geq 0. \end{aligned} \tag{4.17}$$

Just as in the case of the ERC portfolio, the unbounded weights represented by the $\boldsymbol{x}$ vector is normalized between 0 and 1 by dividing each x by the sum of the vector,

$$\boldsymbol{w}_{MP} = \frac{\boldsymbol{x}}{\sum_i^n x_i}. \tag{4.18}$$

4.4.4 Inverse-Variance Portfolio

The inverse variance portfolio is a simplified form of the minimum variance portfolio, disregarding any assumption of correlation between assets. This portfolio can be constructed using an analytical form, following the below given equation

$$\boldsymbol{w}_{IV} = \frac{\operatorname{tr}(\hat{\boldsymbol{\Sigma}}_i)^{-1}}{\sum_i \operatorname{tr}(\hat{\boldsymbol{\Sigma}}_i)^{-1}}, \tag{4.19}$$

where $\operatorname{tr}(\cdot)$ represents the trace operator that isolates the diagonal values of the covariance matrix, measuring only the variance of each asset.

4.4.5 Equally-Weighted Portfolio

The equally-weighted portfolio for n assets is implemented using a simple $1/n$ weighting. For comparison reasons, it uses the same requirement of 24 months of historical data to be eligible for being included in the investment universe n at any given time t.

4.5 Clustering-based Portfolio Optimization Methods

The three included cluster-based allocation methods contain many similar or identical methods and structures. The starting step for all methods includes correlation clustering, with methodology following section 4.3. After this, the methods diverge. HRP and HERC follow a waterfall weight allocation approach, where weights are distributed in a top-down manner using the recursive bisection scheme, further presented in the next sub-section. NCO follow an approach where both between-cluster and within-cluster allocation is determined traditional portfolio optimization methods, broken up into sub-problems, and merged into a final allocation. Below, the implementation for each method are discussed in more detail, and the full implementation in Python is provided in Appendix A3.

4.5.1 Hierarchical Risk Parity

The Hierarchical Risk Parity (HRP) portfolio is applied through a three-step process, following the theoretical framework presented in section 2.8. In contrast to the original implementation, the hierarchical clustering methodology is extended using the optimal leaf ordering algorithm, applied through the Scipy package. Below, each step is outlined in terms of the implementation steps.

Step 1: Hierarchical Clustering

First, the correlation matrix is used to form clusters using the agglomerative hierarchical clustering algorithm, implemented in accordance with section 4.3, including the addition of the optimal leaf ordering algorithm to ensure the assets within the equally-sized clusters are as similar as possible.

Step 2: Matrix Seriation

Second, the asset order implied by the leaf order from the hierarchical clustering is applied to the covariance matrix, reordering the portfolio so that more similar assets are placed closer together.

Step 3: Naive Recursive Bisection

Third, the re-ordered covariance matrix is used in the naive recursive bisection step to allocate weights by recursively bisecting the covariance matrix into equally-sized sub-clusters until each asset is in its own unique cluster, following the steps below.

1. The algorithm is initialized by:
   (a) Set the list of items $C = \{C_0\}$ with $C_0 = \{n\}_{n=1,\dots,N}$
   (b) Initialize asset weights as $w_n = 1, \forall n \in [1, \dots, N]$
2. If $|C_i| = 1 \ \forall C_i \in C$, then stop.
3. For $C_i \in C$ such that $|C_i| > 1$
4. Split C_i into subsets $C_i^1 \cup C_i^2$, where $|C_i^1| = \text{int}[0.5|C_i|]$
   (a) Define the in-cluster weights of $C_i^{(j)}$ as $\boldsymbol{w}_i^{(j)} = \frac{\text{tr}[\hat{\boldsymbol{\Sigma}}_i^{(j)}]^{-1}}{\sum_i \text{tr}[\hat{\boldsymbol{\Sigma}}_i^{(j)}]^{-1}}$
   (b) Define the variance of $C_i^{(j)}$ as $V_i^j = \boldsymbol{w}_i^{(j)\top} \hat{\boldsymbol{\Sigma}}_i^{(j)} \boldsymbol{w}_i^{(j)}$ for $j = 1, 2$
   (c) Compute split factor $\alpha_1 = 1 - \frac{V_i^1}{V_i^1 + V_i^2}$, $\alpha_2 = 1 - \alpha_1$
   (d) Re-scale allocations $w_n \ \forall n \in C_i^1$ by a factor of α_1
   (e) Re-scale allocations $w_n \ \forall n \in C_i^2$ by a factor of α_2
5. Loop step 2

4.5.2 Hierarchical Equal Risk Contribution

The Hierarchical Equal Risk Contribution (HERC) portfolio is applied through a four-step process. It involves the similar process as in the HRP implementation, but with the additional step of implementing the optimal number of clusters, and differs somewhat in terms of the assumptions regarding the recursive bisection.

Step 1: Hierarchical Correlation Clustering

First, the correlation matrix transformed to a distance matrix. The distance matrix is subsequently used to form clusters using the agglomerative hierarchical clustering algorithm, implemented in accordance with section 4.3. The hierarchical structure derived is used in step three to determine the constituents of each sub-cluster.

Step 2: Optimal Number of Clusters

Second, the optimal number of clusters k^* is determined using the Gap statistic, outlined in section 4.3.1. The optimal number of clusters is subsequently used to determine the number of clusters to bisect in the following step.

Step 3: Hierarchical Recursive Bisection

Third, the re-ordered covariance matrix is used in the hierarchical recursive bisection step to allocate weights by recursively bisecting the covariance matrix into sub-clusters according to the binary tree structure until the optimal number of clusters is reached, following the steps provided below.

1. The algorithm is initialized by:
    (a) Set the list of items $C = \{C_0\}$ with $C_0 = \{n\}_{n=1,\dots,N}$
    (b) Initialize asset weights as $W_n = 1, \forall n \in [1, \dots, N]$
2. For $i = 0$ to $k^* - 1$
3. For $C_i \in C$ such that $|C_i| > 1$
    (a) Split C_i into subsets $C_i^1 \cup C_i^2$, where C_i^1 represents the left sub-cluster and C_i^2 represents the right sub-cluster of C_i
    (b) Define the in-cluster weights of $C_i^{(j)}$ as $\boldsymbol{w}_i^{(j)} = \frac{\text{tr}[\hat{\boldsymbol{\Sigma}}_i^{(j)}]^{-1}}{\sum_i \text{tr}[\hat{\boldsymbol{\Sigma}}_i^{(j)}]^{-1}}$
    (c) Define the variance of $C_i^{(j)}$ as $RC_i^j = \boldsymbol{w}_i^{(j)\top} \hat{\boldsymbol{\Sigma}}_i^{(j)} \boldsymbol{w}_i^{(j)}$ for $j = 1, 2$
    (d) Compute split factor $\alpha_1 = 1 - \frac{RC_i^1}{RC_i^1 + RC_i^2}$, $\alpha_2 = 1 - \alpha_1$
    (e) Re-scale allocations W_n $\forall n \in C_i^1$ by a factor of α_1
    (f) Re-scale allocations W_n $\forall n \in C_i^2$ by a factor of α_2

Let $\boldsymbol{W}$ be the vector of weights W_n $\forall n \in [1, \dots, N]$, representing the between-cluster weights for each asset.

Step 4: Within Cluster Weight Allocation

The between-cluster weighted are finally multiplied with the within-cluster weights, computed using the inverse-variance allocation. With this, the weights are ensured to sum to 1. Define the within-cluster weights $\boldsymbol{w_i}$ in cluster C_i $\forall i \in [1, \dots, k^*]$ as

$$\boldsymbol{w}_i = \frac{\text{tr}(\hat{\boldsymbol{\Sigma}}_i)^{-1}}{\sum_i \text{tr}(\hat{\boldsymbol{\Sigma}}_i)^{-1}} \tag{4.20}$$

Let $\boldsymbol{w}$ be the combined vector of weights w_n $\forall n \in [1, \dots, N]$, representing the within-cluster weights for each asset. The final asset weights can be calculated by multiplying the between-cluster weights with the within-cluster weights

$$\boldsymbol{w}_{HERC} = \boldsymbol{W} \cdot \boldsymbol{w}. \tag{4.21}$$

4.5.3 Nested Clustered Optimization

The Nested Clustered Optimization (NCO) portfolio is applied through a four-step process.

Step 1: Hierarchical Correlation Clustering

First, the correlation matrix transformed to a distance matrix. The distance matrix is subsequently used to form clusters using the agglomerative hierarchical clustering algorithm, implemented in accordance with section 4.3. The cluster structure is used in the following step to derive the actual cluster constituents by estimating the optimal number of clusters.

Step 2: Optimal Number of Clusters

Second, the optimal number of clusters is determined using the Gap statistic, outlined in section 4.3.1. The optimal number of clusters ultimately determines the final cluster constituents from which the cluster covariance is derived.

Step 3: Within-Cluster Weight Allocation

Third, the within-cluster weight allocation is determined using the clusters formed in the previous step by creating a covariance matrix for the cluster constituents and optimizing using a given portfolio optimization method. As NCO is considered a framework that can be applied to any given convex optimization-based portfolio allocation method, all methods considered in section 4.4 is applied and tested in the thesis.

Step 4: Between-Cluster Weight Allocation

Fourth, the reduced covariance matrix is determined by only considering the between-cluster variances and covariances. From this, the between-cluster weight allocation is calculated using the same optimization method as in the within-cluster weight allocation step.

4.6 Backtesting Methodology

In this section, the methodology for the empirical tests are presented. A combination of methods are used to provide a transparent and robust backtest.

Using past returns to optimize a strategy generally leads to over-optimistic results since the optimization partly adapts to the particular realization of noise, and is thus unstable over time (Bouchaud & Potters, 2009). The out-of-sample risk of an optimized portfolio is, in reality, greater than the in-sample risk, which itself is an underestimate of the true minimal risk (Bouchaud & Potters, 2009). Furthermore, Arnott, Harvey, and Markowitz (2019) state that one of the worst cases of backtesting error is false positive conclusions, that is, accepting a null hypothesis that does not generalize outside the testing environment.

To reduce the likelihood of finding false positive and inflated results, a Monte Carlo backtesting method is applied in addition to the walk-forward backtest, addressing the limitations of the latter finite backtesting method (López de Prado, 2019b). By running a large number of simulations, the results are less likely to be overfitted to a particular stochastic path as opposed to results solely derived from the standard walk-forward backtest. Through the combination of these two methods, a higher validity of the empirical findings can be achieved by combining asymptotic but stylized results of the Monte Carlo method with finite realistic results from the historical walk-forward backtest.

4.6.1 Monte Carlo Backtest

The Monte Carlo backtest method is implemented for initial backtesting to (1) investigate ways to mitigate the adverse effects of allocation error due to sampling covariance instability, and to (2) investigate the performance between the clustering-based portfolios and the traditional portfolios, with emphasis on the impact of the choice of linkage method. In particular, it is interesting to explore how the potential benefits of hierarchical clustering affect both cases compared to the traditionally constructed portfolios, both in-sample and out-of-sample.

4.6.1.1 Estimation Error in Weight Allocation

The first empirical test follows the methodology outlined in López de Prado (2019a) and the data outlined in section 3.4.1. The goal of the test is to show the effect of estimation noise in the performance of the portfolio optimization methods where hierarchical correlation structures are prevalent.

Using two version of highly stylized block-diagonal correlation matrices presented in 3.4.1, a population covariance matrix $\mathbf{\Sigma}$ can be constructed. From this, the hypothetical true optimized portfolio can be computed, representing the case where no estimation error from noise is present. The weight allocation from the true covariance matrix is then contrasted with the allocations from a large number of simulated sample covariance matrices, $\hat{\mathbf{\Sigma}}$, which are simulated using $N = 100$ random return series over $T = 504$ observations from the true covariance matrix $\mathbf{\Sigma}$ using Gaussian returns.

The above procedure is run over 4,000 simulations for each portfolio, 2,000 using the raw covariance matrix and 2,000 using Ledoit-Wolf shrinkage with constant correlation shrinkage target. This methodology achieves the desired effect of producing significant estimation noise, previously discussed in 2.4.1. Finally, the difference between the weight allocation between the in-sample case and the true case is measured using the *root mean squared error* (RMSE) between the weights

$$\text{RMSE} = \sqrt{\frac{\sum_{t=1}^{T} \left(w_{\hat{\mathbf{\Sigma}},t} - w_{\mathbf{\Sigma},t} \right)^2}{T}} \tag{4.22}$$

4.6.1.2 Risk-Based Out-of-Sample Performance

The second empirical test follows the general methodology by López de Prado (2016) in the original Hierarchical Risk Parity paper with some modifications and additions to create more realistically anchored results. This empirical test aims to provide results over how the in-sample results translate to out-of-sample results, including the impact of the choice of linkage method. This aims to illustrate how well the portfolio captures the generalized patterns of correlation by minimizing the effect of spurious noise on the weight allocation.

A non-parametric generative deep learning method presented by Marti (2019) is used to replicate the highly complex structure of the correlation matrix, not bound by previous historical events. A large number of synthetic correlation matrices are generated using the CorrGAN approach, to be later implemented in the multivariate Gaussian return series, following the process thoroughly described in section 3.4.2. For each iteration, two Gaussian return series following the same correlation structure is created. The first represents the available and observable *in-sample* returns, which are used to allocate weights using the portfolio optimization methods. The second represents the unobserved *out-of-sample* returns that the portfolio optimization method has not seen. From this, portfolio volatilities, diversification ratios, and portfolio weight concentrations of the in-sample and out-of-sample time series of returns are calculated.

4.6.2 Historical Walk-Forward Backtest

The walk-forward historical backtesting methodology follows Coqueret and Guida (2020), where the first step is to split the data into two; the out-of-sample period and the initial buffer period.

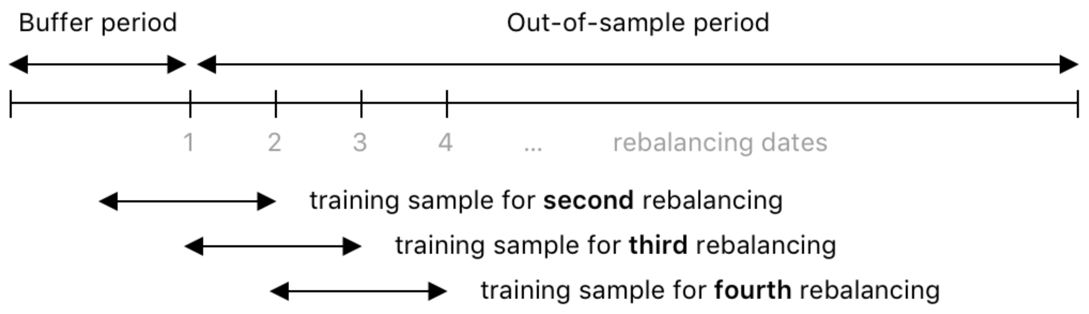
Buffer period
Out-of-sample period
1
2
3
4
...
rebalancing dates
training sample for **second** rebalancing
training sample for **third** rebalancing
training sample for **fourth** rebalancing

Figure 4.1: Historical walk-forward backtesting with rolling rebalancing windows

Source: Coqueret and Guida (2020)

The initial buffer period represents the period from where the first portfolio weights are derived, equal to the length of the covariance matrix estimation. In this paper, this equates to two years ($T = 504$) of daily return data. The out-of-sample returns are calculated by multiplying the returns derived from the training sample using the portfolio optimization method with the returns in the out-of-sample period up until the next rebalancing date. The rebalancing frequency chosen in this paper is once a quarter. The frequency was chosen since this time-frame is widely used in the industry for rebalancing (Tokat & Wicas, 2007).

4.6.3 Statistical Measures

With the risk-based portfolio optimization focus in this study, several statistical risk-based performance measures are presented. This includes the Sharpe ratio, Sortino ratio, Maximum Drawdown, Value-at-Risk, Expected Shortfall, Skewness and Kurtosis, Diversification Ratio, RC Ratio, Sum of Squared Portfolio Weights, and Portfolio Turnover.

4.6.3.1 Sharpe Ratio

A common measure for risk-adjusted returns is the Sharpe Ratio, where a high Sharpe ratio is desirable, all else equal. The *ex-ante* Sharpe ratio is defined as:

$$SR = \frac{\mu - r_f}{\sigma} \tag{4.23}$$

Where μ represents the average return of the asset for the active trading, r_f represents the risk-free rate, and σ represents the standard deviation of the asset's excess return (Sharpe, 1964). Since the population μ and σ has to be estimated, the sample mean and the sample variance is used instead, which results in Lo's (2003) definition of the Sharpe ratio:

$$\widehat{SR} = \frac{\hat{\mu} - r_f}{\hat{\sigma}} \tag{4.24}$$

4.6.3.2 Sortino Ratio

As previously discussed, the Sharpe ratio uses volatility as the proxy for risk, which has its inherent problems. One of these problems is that fluctuations in price, whether they are positive or negative from the mean, are treated equally even though favorable variations are preferred from the investor perspective. As an alternative to the Sharpe ratio, the Sortino Ratio was introduced (Sortino & Price, 1994). The Sortino ratio tries to solve this above-mentioned problem by only using the downside-risk as the proxy for risk instead of the entire asset volatility. The Sortino ratio is defined as

$$\text{Sortino Ratio } = \frac{\bar{r} - r_{MAR}}{\delta_{MAR}} \tag{4.25}$$

where $\bar{r}$ is the expected portfolio return, r_{MAR} is the minimum acceptable return and δ_{MAR} is the downside risk (van der Meer, Sortino, & Plantinga, 2001)

To be able to calculate this ratio, an investor has first to define what is regarded as minimum acceptable return. Any fluctuation above this return is then seen as *positive volatility*, and any return that is below this return is seen as *negative volatility.* This negative volatility is then used to calculate the standard deviation of the downside risk, defined as

$$\delta_{MAR} = \sqrt{\sum_{t=1}^{T} \frac{1}{T} \min\left(0, r_t - r_{MAR}\right)^2} \tag{4.26}$$

In this study, the risk-free rate is used as the minimum acceptable return, $r_{MAR} = r_f$.

4.6.3.3 Maximum Drawdown

Many people consider the maximum drawdown as the most important measurement of risk in a portfolio. Chekhlov, Uryasev, and Zabarankin (2005) defined the maximum drawdown as:

$$\text{Maximum Drawdown} = \max_{\tau \in [0,t]} (W_\tau - W_t) \tag{4.27}$$

Where W_t is defined as the lowest value of the portfolio in the interval $[0, T]$. Important to note from the formula is that the maximum price value has to happen before the drawdown. The maximum relative drawdown instead provide a normalized value between -1 and 0, depicting the percentage loss from the all-time high to the all-time low

$$\text{Maximum Relative Drawdown} = \frac{\max_{\tau \in [0,t]} (W_\tau - W_t)}{W_\tau}. \tag{4.28}$$

This ratio should remain small; otherwise, a portfolio cannot recover. Remember, if the maximum drawdown reaches -50%, the portfolio has to grow 100% to compensate for the previous loss. If we assume that the maximum price of a stock A is 75 USD and that the stock then falls for 1 year down to 5 USD. The maximum drawdown would then be $(75 - 70)/75 = 93\%$. To then recuperate these losses, the stock has to have a $1/(1 - 0.93) = 14.9$ fold price increase. If the annual growth rate is assumed to be 15%, it would take 19 years to get back to the all-time high of 75 dollars. The maximum drawdown does not come without limitations. For example, the maximum drawdown does not take the time or frequency of drawdowns into account. If stock B is assumed to have the same price movement (75 USD to 5 USD), but for 4 months, the maximum drawdown would indicate that these two stocks are equally risky. As such, it is important to be aware of the limitations when comparing this metric over different assets.

4.6.3.4 Value-at-Risk

Value-at-Risk (VaR) evaluates the downside risk of a portfolio as the possible maximum potential change in the value of a portfolio of financial instruments with a given probability over a particular horizon. When assuming normally distributed returns, VaR at a given confidence level $\alpha \in (0, 1)$ can be described as the probability that the loss does not exceed the level inferred by the metric. More formally, it can be defined as

$$\text{VaR}(\alpha) = \mu + \sigma_r \times N^{-1}(\alpha), \tag{4.29}$$

where $N^{-1}(\alpha)$ represents the cumulative probability density function up for the confidence level α. Intuitively, this can be defined as the $(1-\alpha)$ quantile of the return distribution. However, as the distribution of returns does not necessarily follow normal distribution, the VaR can instead be calculated using historical distribution X over a given time as the α quantile of returns over a given time period, defined as

$$\mathrm{VaR}(\alpha, X) = -\inf\{x \in \mathbb{R} : F_X(x) > \alpha\} \tag{4.30}$$

where F_X represents the cumulative distribution of X (Föllmer & Schied, 2011). This method is applied in the paper, as it better captures the risk given the assumption non-normality in the asset return distribution.

4.6.3.5 Expected Shortfall

Value-at-Risk calculates the chance of a loss given a confidence level α. However, it does not conclude anything about the magnitude of losses at investors would like to know what the loss would be in those cases. As a compliment to Value-at-Risk, Expected Shortfall (ES), or sometimes referred to as the Conditional Value-at-Risk (CVaR), can be used to quantify the tail risk of the returns better. Munk (2018) formalize Expected Shortfall as the average of the values beyond the VaR, defined as

$$\mathrm{CVaR}(\alpha) = \frac{1}{\alpha}\int_0^{\alpha} \mathrm{VaR}(\alpha, X)\, dx. \tag{4.31}$$

4.6.3.6 Diversification Ratio

Choueifaty et al., 2013 developed the diversification ratio as a measurement for the diversification in the portfolio. Recall from section 2.7.3; the diversification ratio is defined as the ratio of the portfolios weighted average volatility and its overall volatility

$$DR = \frac{\boldsymbol{w}^\top \boldsymbol{\sigma}}{\sqrt{\boldsymbol{w}^\top \boldsymbol{\Sigma} \boldsymbol{w}}}, \tag{4.32}$$

where $\boldsymbol{w}^\top$ is the vector of asset-weights and where $\boldsymbol{\sigma}$ is the vector of asset volatilities is equal to

$$\sum_{i=1}^{n} w_i \sigma_i = \boldsymbol{w}^\top \boldsymbol{\sigma}. \tag{4.33}$$

The measurement thus quantifies the diversification that is gained by holding assets that are not perfectly correlated (Choueifaty et al., 2013).

4.6.3.7 Risk Contribution Ratio

The risk contribution ratio aims to measure how equally the risk contributions are diversified in the portfolio. This is calculated through the ratio between the portfolio $RMSE$ from the equal-risk contribution case, divided by the same metric calculated using the equally-weighted portfolio (ew). The RC Ratio for portfolio p can thus be calculated as

$$\text{RC Ratio} = \frac{\text{RC RMSE}_p}{\text{RC RMSE}_{ew}} \tag{4.34}$$

where

$$\text{RC RMSE} = \sqrt{\frac{\sum_{t=1}^{T}\left(RC_{it} - \frac{\sqrt{\boldsymbol{w}^\top \boldsymbol{\Sigma} \boldsymbol{w}}}{n}\right)^2}{T}}. \tag{4.35}$$

The denominator is used to make the comparison between portfolio's easier, as both the nominator and denominator will generally be exceedingly small. The risk contribution RC_i for asset i is determined using the same definition as in section 2.7.2.

4.6.3.8 Skewness and Kurtosis

The third and fourth moments of the return distribution, skewness, and kurtosis, can be analyzed to better understand the behavior of the returns, both under normal circumstances, but also in more extreme tail events.

Skewness is a measure of the symmetry in the distribution. A symmetrical dataset has a skewness equal to 0. Essentially, one could say that skewness measures the relative size of the two tails. If a distribution has negative skewness, it implies that there is a greater chance for large negative returns. Wheeler (2011) defines skewness as:

$$a_3 = \sum \frac{(x_i - \bar{x})^3}{ns^3} \tag{4.36}$$

Where n is the sample size, x_i is the i^{th} x value, $\bar{x}$ is the average, and s is the sample standard deviation. Skewness represents the third standardized central moment of the distribution, seen in the exponent of the numerator and denominator in the above expression. To adjust for non-asymptotic sample sizes, the adjusted kurtosis is used

$$\text{Skewness} = \frac{n}{(n-1)(n-2)} \sum \frac{(x_i - \bar{x})^3}{s^3} = \frac{n}{s^3(n-1)(n-2)} (S_{\text{above}} - S_{\text{below}}). \tag{4.37}$$

The difference between these two formulas becomes negligible when the sample size is large. In this paper, the latter definition is used even though the difference is negligible as the sample sizes in most calculations are large.

The fourth moment of the distribution, the kurtosis, can be described as the thickness of the tails. Fat tails imply that the probability of extreme outcomes is higher. Thompson (2013) showed that the return distribution in the market has higher kurtosis due to sudden drops or spikes in prices. This is important to note as a fatter upside implies that there are significant returns on the market that can be captured by an asset manager, while a fatter downside implies that there are larger risks in trading on the market as well (Thompson, 2013). Westfall (2014) defines the kurtosis as:

$$a_4 = \sum \frac{(x_i - \bar{x})^4}{ns^4} \tag{4.38}$$

Where n is the sample size, x_i is the i^{th} x value, $\bar{x}$ is the average and s is the sample standard deviation. The exponent, in this case, is 4 as the kurtosis is often referred to as the "fourth standardized central moment for the probability model" (Westfall, 2014). In practice, the following formula is used in this thesis

$$\text{Kurtosis} = \left\{ \frac{n(n+1)}{(n-1)(n-2)(n-3)} \sum \frac{(x_i - \bar{x})^4}{s^4} \right\} - \frac{3(n-1)^2}{(n-2)(n-3)}. \tag{4.39}$$

This formula does two things differently. First, just as in the case of skewness, it takes the sample size into account. Secondly, it also deducts 3 from the kurtosis, making the central point align with that of the standard distribution. If a distribution has a positive kurtosis, it means that it has fatter tails than a normal distribution. Conversely, if a dataset has negative kurtosis, it means that it has thinner tails than the normal distribution.

4.6.3.9 Sum of Squared Portfolio Weights

The degree of weight diversification is measured using the the squared sum of portfolio weights, presented by Raffinot (2018) as

$$SSPW = \frac{1}{F} \sum_{t=2}^{F} \sum_{i=1}^{N} w_{i,t}^2, \tag{4.40}$$

where F represents the number of portfolio rebalancing periods. The metric ranges from 0 to 1, with lower values representing higher weight diversification. This is also presented by Goetzmann and Kumar (2008) as the approximate deviation from the market portfolio weights, argued as the weight of each security in the market portfolio is very small.

4.6.3.10 Portfolio Turnover

Portfolio turnover is a measure for how often and to what degree assets in the portfolio are being bought and sold. Investors are often interested in portfolio turnover, as trading involves transaction costs. There are many ways of calculating the turnover; in this paper the definition given by Raffinot (2018) is used, defined as

$$TO = \frac{1}{F} \sum_{t=2}^{F} \left| w_{i,t} - w_{i,t-1} \right|, \tag{4.41}$$

where F represents the number of portfolio rebalancing periods. The portfolio turnover is used as a proxy for the theoretical implications of transaction costs on the profitability of the investment strategy. While it is not possible to do an one-to-one comparison, the portfolio turnover still provides valuable insight as a high turnover rate incurs higher costs in comparison to a low turnover rate when evaluating different strategies and portfolios.

5 Empirical Results

In this chapter, the main empirical results are presented. Each of the three tests aims to answer the questions investigated in the thesis. The first test, presented in section 5.1, investigate how and if clustering-based portfolio optimization can improve traditional risk-based portfolio optimization techniques in terms of reduced estimation error. The second test, presented in section 5.2, investigates how the in-sample risk-based performance generalizes to the out-of-sample case of the different portfolios, with emphasis on the impact of different linkage methods. The third and final test, presented in section 5.3, aims to show how the performance translates to real historical data on the S&P 500 in terms of risk-based performance, portfolio concentration, and portfolio turnover.

5.1 Monte Carlo Backtest: Estimation Error

This section presents results from the first Monte Carlo backtest, covering the impact of hierarchical clustering techniques on estimations error. Using the approach described in section 4.6.1 together with the two versions of stylized correlation matrices described in section 3.4.1, a true covariance matrix is constructed. Following this, 100 random multivariate Gaussian return time-series of length 504 are generated, fitted to the true covariance matrix. The sample covariance matrices are then constructed by estimating the covariance of the aforementioned Gaussian returns. The difference between the weights assigned to the sample covariance matrix and the true covariance matrix uncovers the effect of estimation error on the optimization objective. This is calculated using the *root mean squared error* ($RMSE$) of their relative differences.

A Monte Carlo method is used to find a robust estimation of the above problem. The simulation is repeated for 2,000 iterations with and without shrinkage estimation for each optimization objective, including the minimum-variance, equal risk contribution, and maximum diversification. First, the test is run using the block-diagonal correlation matrix to replicate the findings by López de Prado (2019a) in the original NCO paper. Secondly, the test is repeated using a more realistic approach by implementing the stochastic variant, as described in section 3.4.1.

5.1.1 Minimum Variance

In this section, the weight distributions of the MV portfolio is examined together with the NCO portfolio using *minimum-variance* as the objective function.

	MV	**Single Linkage**	**Average Linkage**	**Ward's Method**
Sample Covariance	0.0074	0.0037	0.0037	0.0037
Ledoit-Wolf Shrinkage	0.0065	0.0036	0.0036	0.0036

Table 5.1: $RMSE$ for the long-only minimum variance portfolios using the block-diagonal correlation matrix

From Table 5.1, it is apparent that the original MV portfolio is a less efficient approximation of the true MV weights compared to NCO. The MV portfolio created from the sample covariance matrix shows a $RMSE$ of 0.0074, while the NCO-based portfolios have an estimation error of 0.0037, representing a 50% reduction in the estimation error. These results are in line with the original study performed by López de Prado (2019), demonstrating that the long-only minimum variance portfolio exhibits the same pattern of underperformance in comparison to the equivalent NCO portfolio in terms of estimation error.

When the sample covariance matrix is preprocessed using constant correlation Ledoit-Wolf shrinkage, the empirical results show a uniform improvement as compared to their non-preprocessed counterparts. The minimum variance portfolio is affected the most with a 12% reduction of RMSE, followed by a smaller, but still consistent, reduction of the estimation error of NCO of 4%. These findings follow the fact that the minimum variance portfolio is highly sensitive to estimation errors, but also shows that the NCO is quite resilient in the same regard. Next, the same Monte Carlo simulation is repeated using the randomized block-diagonal correlation matrix.

	MV	**Single Linkage**	**Average Linkage**	**Ward's Method**
Sample Covariance	0.0619	0.0586	0.0585	0.0586
Ledoit-Wolf Shrinkage	0.0619	0.0587	0.0587	0.0587

Table 5.2: $RMSE$ for the long-only minimum variance portfolios using the randomized block-diagonal correlation matrix

As displayed in Table 5.2, the different linkage methods now produce similar but slightly different results, due to the more complex correlation matrix creating a situation where the clustering method sometimes wrongly estimates the optimal number of clusters. In this case, the estimation error, as measured by the $RMSE$, is noticeably higher for all portfolios. The gap between the NCO portfolios and the MV portfolio in estimation error is reduced, with a reduction in estimation error of only 5% as compared to the 50% reduction in the more simplified scenario. This puts the previous findings into question regarding the robustness of the NCO method to achieve lower estimation errors.

5.1.2 Equal Risk Contribution

Next, the ERC portfolio weight distributions together with that of the NCO portfolio using the *equal risk contribution* objective function is examined.

	ERC	**Single Linkage**	**Average Linkage**	**Ward's Method**
Sample Covariance	0.0015	0.0046	0.0039	0.0028
Ledoit-Wolf Shrinkage	0.0014	0.0048	0.0039	0.028

Table 5.3: *RMSE* for the long-only equal risk contribution portfolios using the block-diagonal correlation matrix

Table 5.3 illustrate that the ERC portfolio appears to be quite robust as compared to the MV portfolio, indicated by the comparatively low estimation error of 0.0015 in its original implementation. With this said, the NCO-based method shows little promise for this objective function, as opposed to the case of the MV portfolio, consistently increase the estimation error both with and without shrinkage. This indicates, even using the most basic assumptions, that NCO does not seem to provide any benefit in terms of reduced estimation error for the equal risk contribution objective. To further examine the above results, the same test is run on the stochastic block-diagonal correlation matrix.

	ERC	**Single Linkage**	**Average Linkage**	**Ward's Method**
Sample Covariance	0.0006	0.0054	0.0054	0.0055
Ledoit-Wolf Shrinkage	0.0006	0.0054	0.0054	0.0055

Table 5.4: *RMSE* for the long-only equal risk contribution portfolios using the randomized block-diagonal correlation matrix

As shown in Table 5.4, the estimation error for the ERC portfolio reduced in size, from 0.0015 to 0.0006, providing some empirical grounds for the portfolio being resilient to estimation error when faced with more complex structures. Again, the same pattern is repeated, with a relatively large increase in estimation error for all NCO portfolios as compared to the ERC portfolio. An interesting finding is that shrinkage does not seem to alter the results noticeably for any of the portfolios examined, implying that the portfolio optimization objective of equal risk contribution is reasonably robust to the estimation noise as compared to the MV portfolio, leaving little room for any gains to be made using the shrinkage method applied.

5.1.3 Maximum Diversification

Last, the MD portfolio is examined and compared to the NCO portfolios using the *maximum diversification* objective function.

	MD	**Single Linkage**	**Average Linkage**	**Ward's Method**
Sample Covariance	0.0115	0.0031	0.0031	0.0031
Ledoit-Wolf Shrinkage	0.0102	0.0027	0.0027	0.0027

Table 5.5: $RMSE$ for the long-only maximum diversification portfolios using the block-diagonal correlation matrix

In the results provided in Table 5.5, the MD portfolio exhibits the highest $RMSE$ of all original portfolios examined, indicating a high sensitivity to estimation noise. Again, NCO shows a significant reduction in the estimation error, from 0.0115 in the original MD portfolio to 0.0031 for NCO. This represents a 73.0% reduction in estimation error, the largest gain achieved by NCO compared to its original optimization function in all the tested portfolios.

When the sample covariance matrix is preprocessed using constant correlation Ledoit-Wolf shrinkage, the empirical results again show improvement as compared to their non-preprocessed counterparts. The estimation error of the maximum diversification portfolio is reduced by 11%, while the estimation of NCO is reduced by 13%. The decrease in estimation error comparing MD and NCO suggest that this is the most sensitive optimization method to estimation error. In conclusion, the results indicate that NCO using the long-only, the maximum diversification optimization objective is superior to the traditional long-only MD portfolio in terms of the effects of estimation error to the weight distribution. However, the method is re-examined using the more complex correlation matrix, reported below.

	MD	**Single Linkage**	**Average Linkage**	**Ward's Method**
Sample Covariance	0.0172	0.0115	0.0119	0.0111
Ledoit-Wolf Shrinkage	0.0157	0.0111	0.0116	0.0108

Table 5.6: $RMSE$ for the long-only maximum diversification portfolios using the randomized block-diagonal correlation matrix

Table 5.6 display the results given by the same test performed using the stochastic block-diagonal correlation matrix. Again, the same pattern can be discerned from the previous portfolios, indicating that the reduction in estimation error is diminishing when applying more complex correlation patterns. With this said, there still seems to be some reduction in the estimation error, although less prominent compared to the previous case.

The results from this Monte Carlo backtest applied to the NCO portfolios show inconclusive evidence of any estimation error reduction from hierarchical clustering. The gains are reducing as a function of the complexity in the correlation matrix, putting into question if the results hold up in a realistic setting. This is further investigated in the two subsequent sections.

5.2 Monte Carlo Backtest: Risk-Based Performance

In this section, the performance of the different portfolios is examined, with a focus on investigating the effect of the choice of linkage method in the hierarchical clustering-based portfolios. From a practitioner's viewpoint, it is interesting to see how the cluster-based portfolios perform in the realized out-of-sample case, and how different linkage methods affect the results. Similar to the previous section, a Monte Carlo method is used to investigate to the stated problem.

First, a large number of synthetic correlation matrices are generated using CorrGAN, which are implemented on multivariate Gaussian return series, following the methodology outlined in section 3.4.2. The return series are constructed using 504 return observations, equivalent to 2 years of daily data. Second, the portfolio weight allocations are constructed using the in-sample return series. Third, the weights are then applied to the out-of-sample Gaussian return series over 252 days, equivalent to 1 year of daily return data. Each simulation is run over 2,000 iterations. For the hierarchical clustering-based portfolios, the simulation is executed once for each linkage method discussed in section 2.6.2. Several linkage methods are used to investigate their relative performances, potentially adding to the body of evidence of their performances applied to the different hierarchical clustering-based portfolios.

The main emphasis is placed on the portfolio's objective function metrics, with the other metrics seen as secondary. For the MV portfolio the objective function is represented by the volatility, for the MD portfolio, by the diversification ratio (DR), and for the ERC portfolio, the RC Ratio. In addition, the $SSPW$ is measured to provide intuition about the portfolio weight concentration in the respective portfolios.

5.2.1 Minimum Variance

First, the results for the minimum-variance portfolio and the NCO portfolio using the minimum-variance objective function are presented.

	MV	**Single Linkage**	**Average Linkage**	**Ward's Method**
In-Sample Volatility	10.3%	10.4%	10.8%	10.8%
Out-of-Sample Volatility	10.6%	10.7%	11.1%	11.1%
In-Sample DR	2.20	2.15	2.13	2.13
Out-of-Sample DR	2.14	2.11	2.10	2.09
In-Sample RC Ratio	4.5	4.3	4.0	4.0
Out-of-Sample RC Ratio	4.6	4.4	4.0	4.0
SSPW	0.12	0.11	0.10	0.10

Table 5.7: Performance Statistics for Minimum-Variance Portfolios

As shown in Table 5.7, the in-sample volatility for the Single Linkage portfolio is the lowest observed among the NCO portfolios, with a value of 10.4% annualized in-sample volatility, increasing to 10.7% out-of-sample. This can be compared to the original MV portfolio with a slightly lower out-of-sample volatility of 10.6%. For Average Linkage and Ward's Method, the volatility out-of-sample were both 11.1%.

In terms of DR, the MV portfolio slightly outperformed the NCO portfolios, both in-sample and out-of-sample. The observed DR for the NCO portfolios exhibit very similar ratios, with somewhat better performance of Single Linkage. The results illustrate that all of the NCO portfolios decrease the RC Ratio, indicating that the equality in risk contribution is somewhat increased, regardless of the linkage method applied. The results of the $SSPW$ metric shows that the weight distribution is most concentrated in the original portfolio, with consistently better performance from all different NCO portfolios, with the most diversified portfolio being produced using Ward's Method.

In summary, the results show that all portfolios exhibit similar performance in terms of the portfolio risk metrics discussed, with slightly better results from the original MV implementation. However, as some of the best results are found in the NCO portfolios, further investigation into the risk-based performance of the NCO is warranted using historical data.

5.2.2 Equal Risk Contribution

In this section, the results of the ERC portfolio together with the NCO with the equal risk contribution objective function is examined.

	ERC	**Single Linkage**	**Average Linkage**	**Ward's Method**
In-Sample Volatility	17.0%	16.8%	16.0%	17.1%
Out-of-Sample Volatility	17.0%	17.1%	16.3%	17.2%
In-Sample DR	2.02	2.06	2.13	1.99
Out-of-Sample DR	2.02	2.03	2.11	1.99
In-Sample RC Ratio	0.0	7.0	3.2	1.4
Out-of-Sample RC Ratio	0.3	7.1	3.3	1.4
SSPW	0.01	0.10	0.04	0.02

Table 5.8: Performance Statistics for Equal Risk Contribution Portfolios

The goal of the equal risk contribution portfolio is to minimize the RC Ratio. As displayed in Table 5.8, the traditional ERC portfolio performed better than its NCO counterparts, displaying an out-of-sample RC Ratio of 0.3. In terms of the NCO portfolios, the Ward's Method produced the lowest RC Ratio, while Single Linkage produced the highest value. The discrepancy between the values indicate that the choice of linkage method affects the target objective to a large degree, with the out-of-sample RC Ratio value varying between 1.4 and 7.1.

The ERC portfolio display a volatility of 17.0%, both in-sample and out-of-sample. From this perspective, the Average Linkage portfolio performed best out of all portfolios, with somewhat lower volatilities. The other linkage methods display similar performance to the case of the original ERC portfolio. However, in terms of weight diversification according to the $SSPW$ metric, the original ERC portfolio shows a better diversification, closely followed by Ward's Method and Average linkage. Thus, all NCO portolios underperform in terms of the objective function of equal risk contribution, and seem to produce more concentrated portfolios. In this case, Ward's Method produced the results closest to the ERC portfolio, with fairly similar results across the board.

5.2.3 Maximum Diversification

Next, the MD portfolio is examined and compared with the NCO counterparts using the maximum diversification objective function.

As seen in Table 5.9, the portfolio with the highest DR is the traditional MD portfolio, both

	MD	**Single Linkage**	**Average Linkage**	**Ward's Method**
In-Sample Volatility	13.1%	16.0%	15.3%	15.7%
Out-of-Sample Volatility	13.7%	16.5%	15.7%	16.0%
In-Sample DR	2.67	2.14	2.25	2.17
Out-of-Sample DR	2.51	2.09	2.20	2.13
In-Sample RC Ratio	3.9	8.3	5.5	3.7
Out-of-Sample RC Ratio	4.1	8.4	5.7	3.7
SSPW	0.07	0.13	0.07	0.04

Table 5.9: Performance Statistics for Maximum Diversification Portfolios

in-sample and out-of-sample. Out of the clustering-based portfolios, the Average Linkage portfolio shows the most promising results, both in terms of volatility and DR, however, still consistently less so than the traditional MD portfolio. Ward's Method produce somewhat higher volatility but looks to reduce the weight concentration and increase the RC Ratio, both relative to the other NCO portfolios and the MD portfolio.

The above findings indicate that NCO consistently fails to provide lower risk and diversification in the case of the maximum diversification, albeit with somewhat more weight diversification and even risk contribution resulting from Ward's Method. This comes at no surprise as the objective function aims to find the most diversified portfolio among all assets, implying that a larger asset universe is more likely to contain assets with less correlation. The empirical results further back this point as the more balanced linkage methods of Average Linkage and Ward's Method resulted in higher diversification metrics. In summary, Average Linkage exhibits the highest DR, while Ward's Method exhibits the lowest weight concentration and most equal risk contribution.

5.2.4 Hierarchical Risk Parity

In this section, the results for the HRP portfolios are presented and contrasted to the closest non-hierarchical counterpart, the IV portfolio. The HRP portfolio is mainly assessed in terms of the volatility and the RC Ratio.

As shown in table 5.10, the best performing portfolio both in terms of volatility and DR is the Single Linkage portfolio, in agreement with the existing evidence gathered by (Raffinot, 2018). It displays an in-sample volatility of 15.1%, increasing to 15.2% out-of-sample, indicating that the performance generalizes from in-sample to out-of-sample performance quite well. However, the results from the other linkage methods are very close, with Average linkage and Ward's

	IV	**Single Linkage**	**Average Linkage**	**Ward's Method**
In-Sample Volatility	16.1%	15.1%	15.2%	15.2%
Out-of-Sample Volatility	16.1%	15.2%	15.2%	15.3%
In-Sample DR	1.92	2.05	2.04	2.04
Out-of-Sample DR	1.92	2.04	2.04	2.03
In-Sample RC Ratio	0.8	3.6	3.6	3.6
Out-of-Sample RC Ratio	0.9	3.5	3.6	3.5
SSPW	0.01	0.02	0.02	0.02

Table 5.10: Performance Statistics for Hierarchical Risk Parity Portfolios

Method producing close to identical results in terms of all metrics used. Since HRP only relies on hierarchical clustering to determine the order of the assets, one can infer that these two linkage methods produce similar orderings, and thus, similar results.

The weight concentration and RC Ratio inferred show no clear difference between the implementations, indicating that the weight concentration produced is independent of the linkage method. This can be attributed to the fact that the recursive bisection is allocating to equally-sized sub-cluster, independent of the hierarchical structure inferred by the linkage method. The weight concentration is, however, somewhat higher than that of the non-hierarchical IV portfolio, indicating that even the most symmetrical of recursive bisection methods induce some concentration to the weight allocation, all else equal.

5.2.5 Hierarchical Equal Risk Contribution

In this section, the results for the HERC portfolios are examined and contrasted to the closest non-hierarchical counterpart, the IV portfolio. Like HRP, the HERC portfolio is referred to as a risk parity portfolio. However, it also use the inverse-variance allocation, indicating a goal of minimizing the variance. Therefore, the most relevant metrics are assessed to be both the volatility and the RC Ratio.

Table 5.11 indicates that all of the HERC portfolios seem to perform on a similar level. The volatility is slightly lower for Average Linkage, but the RC Ratio is lowest for Ward's Method. In terms of risk contributions, these findings are in line with the original study by Raffinot (2018). The results indicate that the HERC portfolios consistently outperform the benchmark IV portfolio in terms of volatility and diversification ratio. However, the IV portfolio show lower weight concentrations and more even risk contributions.

	IV	Single Linkage	Average Linkage	Ward's Method
In-Sample Volatility	16.1%	15.2 %	14.9 %	15.5 %
Out-of-Sample Volatility	16.1%	15.5 %	15.0 %	15.5 %
In-Sample DR	1.92	2.07	2.08	2.00
Out-of-Sample DR	1.92	2.04	2.07	2.00
In-Sample RC Ratio	0.8	6.0	2.3	1.3
Out-of-Sample RC Ratio	0.9	6.0	2.3	1.3
SSPW	0.01	0.09	0.03	0.02

Table 5.11: Performance Statistics for Hierarchical Equal Risk Contribution Portfolios

Again, the recursive bisection seems to induce higher weight concentrations compared to the case of the IV portfolio. In this case, and in contrast to the HRP case, the concentration can be linked to the linkage method, indicating that the dendrogram used in the recursive bisection indeed has major significance in the weight concentration. The results are somewhat inconclusive of the effectiveness of HERC to achieve the different objectives, and a more in-depth testing using historical data is warranted.

5.3 Historical Backtest

In this section, the results of the historical walk-forward backtest on the S&P 500 investment universe are presented. All dividend proceedings are assumed to be reinvested, and the performance is evaluated in terms of risk-based performance metrics, portfolio weight concentration, and portfolio turnover. Furthermore, Ward's Method is applied as the linkage method for the cluster-based portfolios, with the exception of the HRP portfolio, using Single Linkage following the methodology by López de Prado (2016). This will increase the comparability between the NCO portfolios, and reduce the number of portfolios used, decreasing potential data mining bias. The results from other linkage methods are included in Appendix A5, but are not further discussed in this section.

5.3.1 Returns & Volatility

This section presents the observed returns and volatility of the tested portfolios. Figure 5.1 presents the cumulative return of the different portfolios. Initially, it can be observed that the cumulative return of the MD portfolio is significantly higher than the rest of the portfolios, followed by the NCO MD counterpart. These portfolios are the only one's that produce higher returns than the EW benchmark in this setting. These results are in line with previous findings by (DeMiguel et al., 2009), illustrating that under most circumstances, the EW portfolio is hard to beat on a pure return basis. Furthermore, the results suggest that the MV portfolio and the NCO MV portfolio are producing the lowest cumulative returns over the period, but also seem to exhibit the lowest volatility. The ERC, NCO ERC, IV, HRP, and HERC portfolios show similar performance, and all fall in-between the EW and MV portfolios.

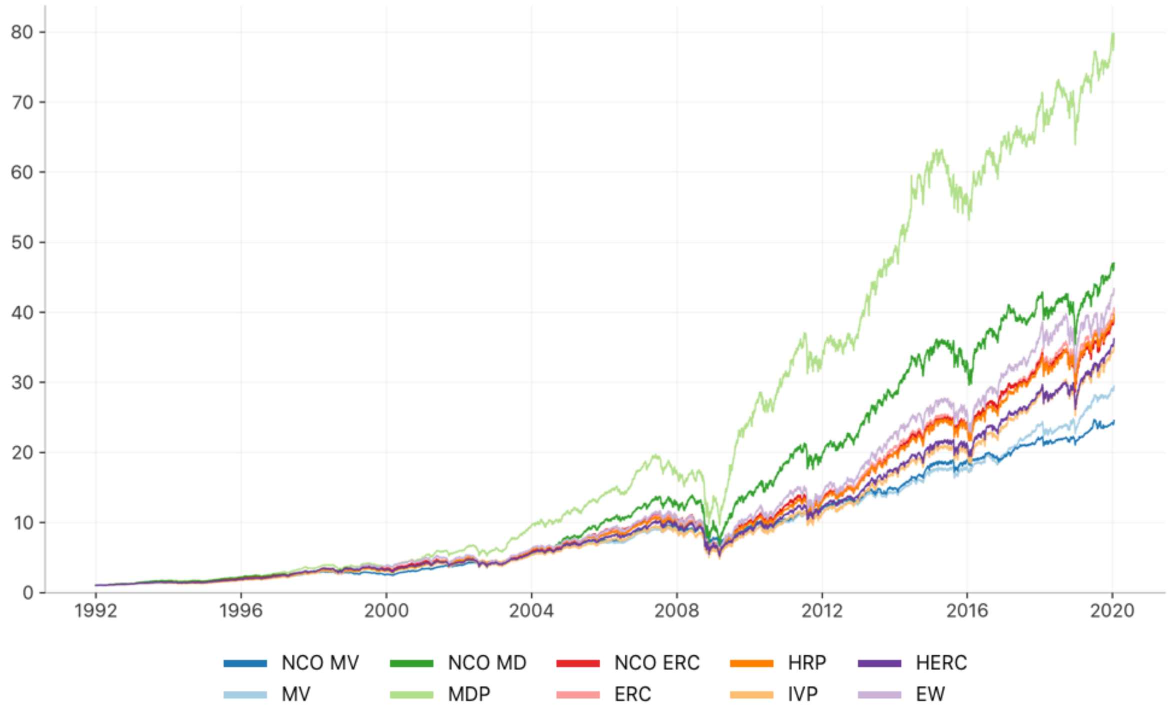

Figure 5.1: Cumulative Portfolio Returns

As previously stated, Figure 5.1 does not provide a full picture of the assessed performances of the different portfolios, as the returns are not a part of the portfolio optimization objective. As such, the subsequent sections dig deeper into the risk-based characteristics of the portfolios.

In Table 5.12, it can be inferred that the NCO MV portfolio and the MV portfolio display the lowest yearly volatility. This indicates that they both are efficient in terms of allocating weights to achieve the lowest variance, even out-of-sample. However, the NCO MV portfolio performs worse in terms of the risk-adjusted returns, as the yearly return is lower in comparison to the traditional MV portfolio while displaying almost identical volatility. A similar pattern can be discerned for the MD and NCO MD portfolio, with the NCO MD portfolio lower returns the traditional counterpart, while also demonstrating higher volatility. The same is also true in the case of ERC, where the NCO and the traditional portfolio display the same levels of volatility

	Yearly Return	**Yearly Volatility**
EW	16.3%	18.1%
MV	13.4%	10.0%
NCO MV	12.7%	10.0%
MD	18.0%	13.5%
NCO MD	16.1%	15.2%
ERC	15.6%	15.9%
NCO ERC	15.4%	15.9%
IV	14.8%	15.1%
HRP	15.2%	14.4%
HERC	14.8%	14.3%

Table 5.12: Return and Volatility of Portfolios

but somewhat lower returns. In summary, there seems to be no noticeable reduction in the volatility of the NCO portfolios, a finding along the lines with the results from the Monte Carlo backtest performed in section 5.2. The HRP and HERC portfolios exhibit almost identical volatility estimates, both of which outperform the IV benchmark with a noticeable reduction in the volatility, in line with previous findings of the Monte Carlo backtest.

Figure 5.2 provides a more in-depth view of the portfolio volatilities using a 24-month rolling average. As shown, the distribution of the volatility over time seem to consistently favor the minimum-variance portfolios, along with the non-hierarchical MD portfolio. It comes as no surprise that all portfolios display an increase in volatility during times of financial distress, and the results indicate that the volatility spread tends to increase between the portfolios during these conditions. This is most noticeable in the volatility spike during the Financial Crisis around 2008 up until 2011, where the volatility of the MV portfolios is significantly lower in comparison to the other portfolios, providing further evidence that it generally produces low-variance results.

Furthermore, as seen from Figure 5.2, no general difference between the NCO portfolios and their traditional counterparts can be discerned, implying that there is little benefit to choosing one over the other in this regard. Again, HRP and HERC show a pattern of consistently outperforming the IV portfolio in terms of volatility. This indicates that the hierarchical clustering together with the recursive bisection method seems to benefit from the implied hierarchical structure of the assets in reducing volatility, even in distressed market conditions.

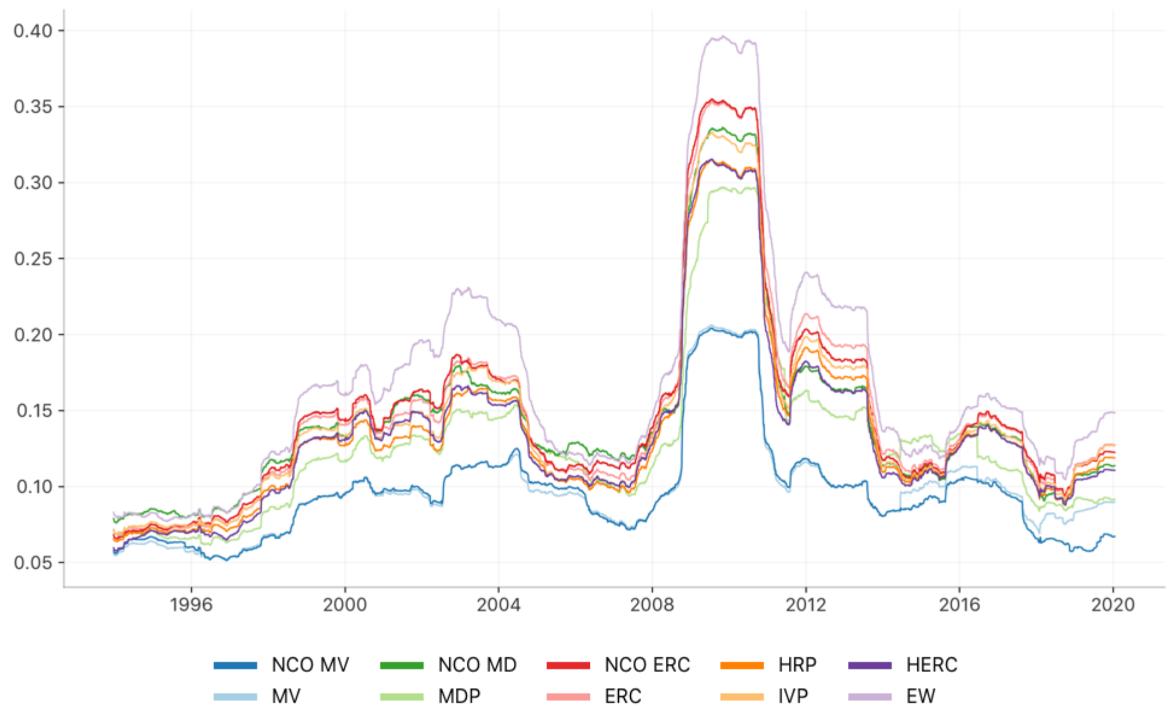

Figure 5.2: 24-Month Rolling Volatility

5.3.2 Daily Return Distribution & Descriptive Statistics

In this section, the characteristics of the returns for the different portfolios are presented, providing a deeper picture of the return distribution and its implication of the portfolio risk.

Figure 5.3 shows the distribution of returns for the different portfolios. The results indicate that the portfolios are fairly similar in terms of their return distribution across the board. The minimum variance portfolios are clear outliers, depicted by lower volatility as compared to the other portfolios, meaning that more of the return probability distribution is located near the center. On the other end of the spectrum, the EW portfolio shows the widest return distribution, and in between these two ends, the other portfolios show fairly similar distributions.

Table 5.13 provide detailed results over the descriptive statistics of all portfolios. The MV portfolios exhibit the smallest percentile daily movement, further validating that they are fulfilling their objective of producing low variance results, regardless of the direction of the return movement. The kurtosis levels of the portfolios ranges between 6.53 and 13.11. The highest kurtosis is can be found in the MD portfolio, indicating higher probability of extreme return events. On the other end, the low kurtosis of the EW portfolio indicates less tail probability. However, the relatively high volatility of said portfolio makes it hard to compare to the other

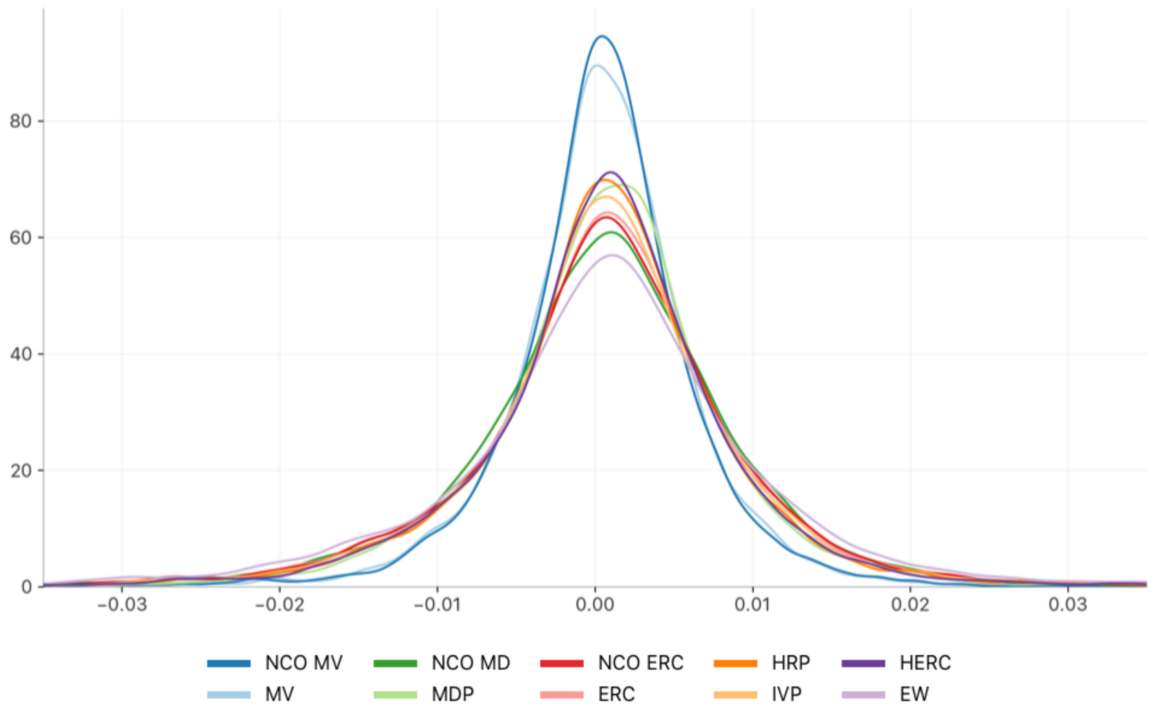

Figure 5.3: Distribution of Daily Returns

portfolios in absolute terms, as the kurtosis is highly dependent on the level of the volatility.

The IV, HRP, and HERC portfolios display similar results in all metrics of the descriptive statistics, suggesting that both HRP and HERC retain much of the same base characteristics as the IV while providing lower volatility, as shown in Table 5.12. All portfolio display a skewness between -0.25 and 0.22, illustrating that the daily returns for all different portfolios are fairly symmetrical. The MD and MV portfolios are the only two displaying a positive skewness, indicating slightly more probability of positive than negative returns on a daily basis.

5.3.3 Risk-Based Performance

In this section, the risk-based performance measures for each of the portfolios included are examined using the metrics provided in Table 5.14, including the maximum drawdown, the annualized VaR (5%), the annualized CVaR (5%), the Sharpe ratio, and the Sortino ratio.

The two MV portfolios exhibit lower downside risk as measured by the maximum drawdown, VaR (5%), and CVaR (5%). These metrics indicate lower values for all metrics with a wide margin. On the other end, the EW portfolio performed worst across the same metrics, highlighting the

	Skewness	Excess Kurtosis	Max	Min	5th percentile	95th percentile
EW	-0.19	6.53	11.13%	-9.94%	-1.68%	1.65%
MV	0.09	12.39	8.74%	-6.27%	-0.93%	0.98%
NCO MV	-0.11	11.33	8.72%	-6.50%	-0.91%	0.96%
MD	0.22	13.11	12.77%	-7.92%	-1.22%	1.26%
NCO MD	-0.21	7.51	11.38%	-8.68%	-1.41%	1.40%
ERC	-0.21	8.15	10.89%	-9.10%	-1.47%	1.42%
NCO ERC	-0.25	10.07	12.46%	-9.85%	-1.45%	1.41%
IV	-0.16	8.51	10.69%	-8.71%	-1.40%	1.35%
HRP	-0.20	8.35	10.35%	-8.25%	-1.35%	1.29%
HERC	-0.24	9.88	11.02%	-8.63%	-1.31%	1.30%

Table 5.13: Distributions of Portfolio Returns

	Max Drawdown	VaR (5%)	CVaR (5%)	Sharpe Ratio	Sortino Ratio
EW	-55.7%	26.6%	43.1%	0.67	0.97
MV	-30.2%	14.0%	23.0%	0.88	1.31
NCO MV	-31.3%	14.4%	23.2%	0.88	1.29
MD	-51.2%	19.3%	30.6%	0.99	1.47
NCO MD	-50.6%	22.5%	35.9%	0.77	1.11
ERC	-52.5%	23.3%	37.9%	0.72	1.04
ERC NCO	-53.9%	23.0%	37.7%	0.71	1.03
IV	-49.9%	22.3%	36.1%	0.72	1.05
HRP	-49.3%	21.5%	34.4%	0.78	1.12
HERC	-49.6%	20.9%	34.1%	0.75	1.09

Table 5.14: Risk-based performance

fact that low weight concentration does not necessarily equate to high risk diversification. The MD portfolio exhibits the highest value for both the Sharpe ratio and the Sortino ratio, with values of 0.99 and 1.47 respectively. Although the NCO MD provided the second highest returns, the returns seem to have a proportional response in the risk, indicated by the comparably lower estimates of the Sharpe and Sortino ratios of 0.77 and 1.11. Furthermore, the MV and NCO MV portfolios display comparably high Sharpe and Sortino ratios, implying that the risk-based returns for these two portfolios are competitive, even after considering their relatively low returns. The HRP indicated the second-best Sharpe and Sortino ratios of the clustering-based portfolios, followed closely by NCO MD, and HERC. However, the metrics are fairly similar, with the Sharpe and Sortino ratio between 0.71 to 0.78, and 1.02 to 1.12, respectively.

Finally, when assessing the risk-based performance metrics above, it is evident that the EW portfolio's comparably high returns come at a cost, as it consistently displays poor performance across all metrics, outperformed by all the included portfolios across the board. This indicates that all portfolios constructed provide better risk-adjusted performance as compared to the most naive solution of equal weighting.

5.3.4 Portfolio Weights and Portfolio Turnover

In this section, the portfolio weight distribution and the portfolio turnover characteristics are examined in detail. This is an important aspect as portfolios with good risk-based results might still prove impractical to implement due to prohibitive high weight concentrations or high transaction costs from portfolio turnover between rebalancing periods.

	Portfolio Turnover	**SSPW**	**Avg. Max Weight**
EW	2.6%	0.0017	0.2%
MV	46.5%	0.0593	12.3%
NCO MV	61.4%	0.0477	10.5%
MD	49.2%	0.0357	8.4%
NCO MD	111.8%	0.0228	5.9%
ERC	6.6%	0.0021	0.69%
NCO ERC	45.9%	0.0034	1.2%
IV	7.7 %	0.0027	0.9%
HRP	26.7%	0.0034	1.7%
HERC	47.8%	0.0050	2.1%

Table 5.15: Portfolio Weights and Portfolio Turnover Metrics

Table 5.15 indicate that both the MV portfolios and the MD portfolios are holding few assets, concentrating much of the weight to a small subset of assets, a finding confirming previous findings of (Choueifaty et al., 2013). This is highlighted by the weight concentrations shown by $SSPW$ together with the average maximum weight an a single asset. The MV and the NCO MV portfolios have an $SSPW$ of 0.0593 and 0.0477 respectively, the largest among the portfolios tested.

The HRP and the HERC portfolios produce among the lowest $SSPW$, in accordance with previous results by Raffinot (2018), indicating that these portfolios produce relatively well-diversified portfolios in terms of weight concentration. However, they all show higher values than the ERC, NCO ERC, and IV portfolios, backed by findings in the Monte Carlo backtest presented in section 5.2. In addition, the results imply NCO accomplishes lower weight concentrations compared to their MV and MD counterparts. Again, validated by the Monte Carlo backtest, indicating that hierarchical clustering using Ward's Method can potentially be used to decrease weight concentration for portfolios with high weight concentrations.

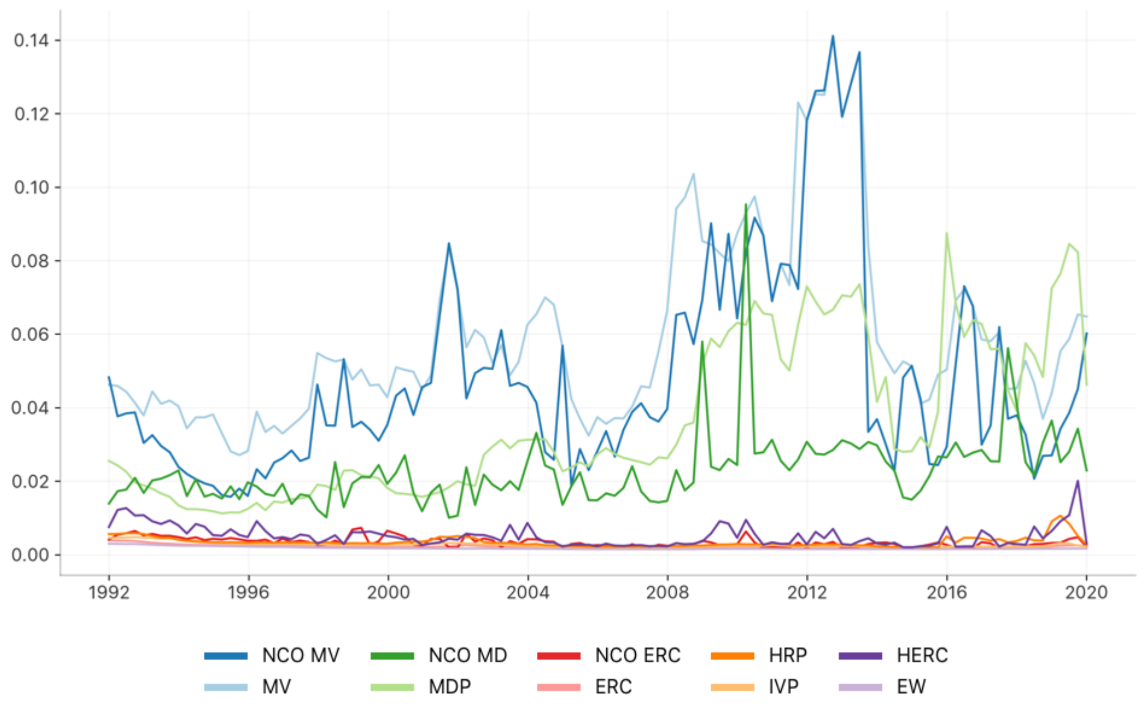

Figure 5.4: Sum of Squared Portfolio Weights per Rebalancing Period

Figure 5.4 provide the sum of squared portfolio weights, $SSPW$, depicted at each rebalancing period. This highlights the stability of weight concentrations in the portfolios. Again, NCO show promise in providing lower weight concentrations for the NCO portfolios in comparison to their counterparts for the minimum variance and maximum diversification objectives, albeit with somewhat higher intertemporal fluctuations.

In Table 5.15, the results indicate that all NCO-based portfolios produce higher average portfolio turnover in comparison to their counterparts. This is true in all cases, where the portfolio turnover increase close to between 20% and 60% respectively per rebalancing when applied using the NCO method. As the portfolio's in this backtest are rebalanced quarterly, the difference in their yearly portfolio turnover can be considered quite substantial. A large portfolio turnover is a problem as it indicates that the cost of maintaining the portfolio is higher than for comparable portfolios. Of the clustering-based portfolios, the HRP portfolio exhibits the lowest portfolio turnover, followed by NCO ERC and HERC.

As seen in Table 5.15, the EW, ERC, and IV portfolios exhibit the lowest levels of average turnover, indicating stable weight allocations. Of the traditional portfolios, MV and MD display the highest levels of turnover. Considering this, their strong risk-based performance does seem to come at a cost of higher portfolio turnover.

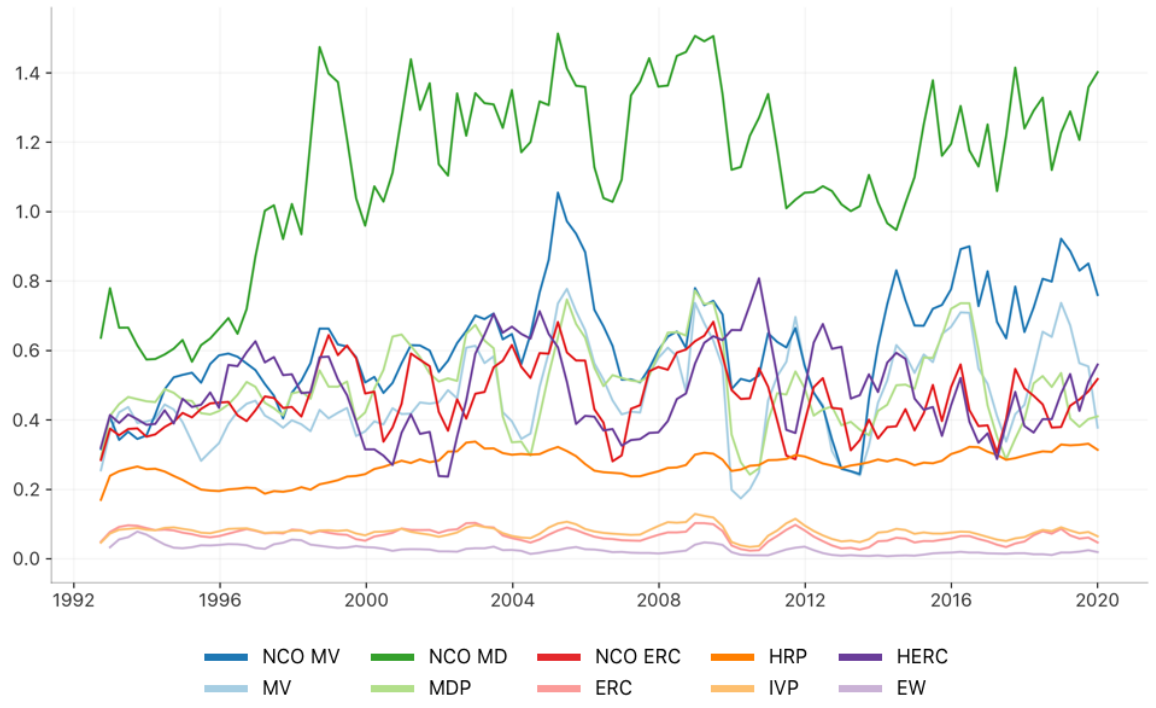

Figure 5.5: 12-Month Rolling Average Portfolio Turnover

Figure 5.5 shows the portfolio turnover as a function of time. This display how the turnover differs among the portfolios terms of their level and variability. On the more stable and low end, the EW, ERC, and IV portfolios can be found. These represent the most intertemporally stable portfolios in terms of weight allocations, also supported by their low weight concentrations. HRP show the same level of stability in turnover over time, but at a consistently higher level. However, the level is still consistently lower than that of the other clustering-based portfolios.

On the other end, the portfolio with the most volatile and largest portfolio turnover is the NCO MD portfolio, followed by the NCO MV portfolio. These portfolios show volatile behavior over time, indicating that even if they were to provide good risk-based performance, some hesitation should be taken before applying these in practice. In between these extremes, the NCO ERC, MV, MD, and HERC portfolios show somewhat volatile intertemporal turnover.

6 Discussion

In this chapter, the findings from the empirical results are discussed to provide insights into the implications and practical applicability of the methods researched. The following sections are structured according to the different backtests, thus following the same structure as in chapter 5. Finally, the practical implications and proposed topics of further research are discussed to provide an analysis of the strengths and weaknesses of the methods for the use in practical portfolio allocation scenarios.

6.1 Monte Carlo Backtest: Estimation Error

From the empirical test following the methodology of López de Prado (2019), there initially seems to be significant gains to be made from clustering in reducing the estimation error in portfolio optimization. For the NCO portfolios using the minimum-variance and maximum diversification objectives, the results are at first promising. However, it is hypothesized that much of the decrease in estimation error can be explained by the simple stylized correlation matrix not corresponding to a realistic scenario. As such, estimation noise in the correlation matrix together with non-uniform and non-zero correlations would make it harder for the hierarchical clustering algorithm to find the optimal number of clusters and increase the condition number of the reduced covariance matrix. This suspicion is confirmed when applying the more irregular, and somewhat more realistic, stochastic block-diagonal correlation matrix. All portfolios tested show significantly higher estimation errors as measured by the $RMSE$. Besides, the gap in performance between the original portfolios compared with their NCO counterparts is almost closed, indicating small to non-existent gains from these portfolios in terms of estimation errors. This implies that the more simplified assumption used in the original study by López de Prado (2019a) produce preferable results in favor of the clustering-based methods, possibly explained by two factors.

First, as the between-cluster correlation is zero, the clusters are well separated and easily distinguishable for the clustering algorithm. Indeed, as reported in the empirical results, all different linkage methods successfully find the correct number of clusters in all of the iterations of the simulation. This is confirmed as all linkage methods produce identical results, indicating that they have no trouble finding the correct number of clusters.

Secondly, the same assumption about between-cluster correlation produces a close to diagonalized reduced covariance matrix for the between-cluster weight allocation step in the NCO. This would typically not be the case in a real setting, as there are significant positive correlations present between clusters. In other words, the gains made in terms of reduced estimation error from the structure of the correlation matrix, are theorized to be vastly overstated in this stylized example. Again, this is confirmed when applying the more complex stylized correlation matrix, indicating that less reduction in the condition number can be achieved than first anticipated.

In summary, the empirical findings show how hierarchical clustering theoretically can reduce estimation error by breaking the optimization problem up in several more well-behaved sub-problems, given the right circumstances. However, as explored further using the more realistic stochastic block-diagonal correlation matrix, this pattern does not seem to hold up very well, suggesting that the application of clustering to reduce estimation error lack robustness in its current form. Besides, results from the other empirical tests in the thesis provide further evidence that this is true, discussed in detail in the next two sections.

6.2 Monte Carlo Backtest: Risk-Based Performance

The Monte Carlo backtest applied to infer the in-sample and out-of-sample performance of the different portfolios provides further empirical evidence that NCO shows little to no additional gains in regards to several risk-based metrics applied. From the results, it is hard to infer that any specific linkage method is superior, and the different methods seem to fit better with different optimization objectives. This can be attributed to the nature of the linkage methods, where Ward's Method produces the most well-balanced clusters, and Single Linkage produces the least well-balanced clusters. As the between-cluster weight allocation does not account for the difference in the relative size of the clusters, more uneven clusters produced by the clustering algorithm likely result in more dense weight concentrations in the portfolio. This pattern is easily discerned, where linkage methods with more chaining behavior produce more concentrated weight allocations, whereas more balanced ones produce well-balanced weight allocations. This finding is further supported by previous results shown by Papenbrock (2011). An exception to this rule can be found regarding the HRP portfolio due to its implied dendrogram structure being independent of the linkage method applied. As such, the HRP portfolio always produces perfectly well-balanced clusters. The empirical findings in section 5.2 support this as it exhibits low portfolio concentration as measured by $SSPW$, regardless of the linkage method applied.

The choice of linkage method can thus arguably be seen as a strategic issue and should be chosen with considerations to the combined choice of portfolio optimization method and strategic objective of the portfolio allocation. As explored by Papenbrock (2011), Single Linkage can be applied opportunistically and potentially gain from a few diversified stocks. On the other end, Ward's Method can be applied to produce lower weight concentrations, and thus more stable results. In between these extremes, Average Linkage can be applied as a middle-ground solution. Both the HRP and HERC portfolios show promise in reducing the risk and allocating to more diversified sources as compared to the inverse-variance portfolio but does not seem to hold up in comparison to many of the convex optimization solutions, with consistent underperformance in regards to the risk-based performance metrics applied in this thesis.

The empirical findings from this Monte Carlo backtest show how the linkage method used can affect the outcome of both the risk-based performance in terms of the portfolio's optimization objective, but also the weight concentration. For a more in-depth analysis, the same methods were applied to the historical backtest, discussed more in the next section.

6.3 Historical Backtest

The findings in the historical backtest further support the results of the Monte Carlo backtest, again indicating that the potential gain from NCO is limited. Any increase in the risk-based performance made is considered either small or unstable, implying that this method does not seem to provide an increase in the assessed performance with any degree of certainty. López de Prado (2019a) suggests to apply and use NCO as a general method to increase the performance in portfolio optimization. The findings in section 5.3 provide further empirical evidence contradicting this, as the NCO exhibits unstable allocations and no clear pattern of any risk-based performance increase. In the historical backtest, the NCO portfolios are at best performing at the same level as their traditional counterpart, or at worst considerably worse than its counterpart.

The results for the HRP and HERC portfolios are in line with the results from the previous Monte Carlo backtest and Raffinot (2018), further validating the findings that these methods indeed utilize the hierarchical structure to reduce risk in comparison to the IV portfolio. In comparison to the IV and ERC portfolios, both the HRP and HERC portfolios exhibit higher Sharpe and Sortino ratios and also indicate lower VaR, CVaR and Maximum Drawdowns. As such, this shows some promise in their relative performances, but as discussed below, the impact of weight concentration and portfolio turnover has to be reviewed to get a more nuanced image of their applicability.

When accounting for portfolio turnover between rebalancing periods, the findings show that the practical applicability of the clustering-based methods is somewhat limited using the methodology applied in this thesis, especially true for the NCO method. Regardless of the linkage method applied, the clusters seem to be more or less unstable over time, implied by the substantial weight differences between rebalancing periods. This is evident when looking at the portfolio turnover of the NCO and HERC portfolios compared to their benchmarks, putting into question if the potential gain in risk-based performance can justify the increase in transaction costs. It remains unclear if more stable correlation clusterings can be achieved by using another clustering approach or another covariance matrix pre-processing method, a topic proposed to be further investigated in future research.

In the end, the empirical results from both the historical backtest and the second Monte Carlo backtest show that the traditional long-only minimum-variance, equal risk contribution, and maximum diversification portfolios perform best out-of-sample. Both in terms of their investment objectives, but also with regards to many of the risk-based statistical performance measures.

6.4 Practical Applications

From the different tests conducted in section 5, there seems to be little to no benefit of applying the hierarchical portfolios from a risk-based perspective when accounting for the portfolio turnover incurred. The portfolios outperformed the traditional portfolios in section 5.1, but when applying more realistic assumptions, these results quickly diminished. From this, the estimation error reduction is hypothesized to be diminishing when moving closer to the highly complex structure that is found in a correlation matrix produced from real financial data. This is supported by the findings from the two subsequent tests, using data with less well-formed clusters and with real historical data. The clustering-based portfolios might still reduce the estimation error, but to a smaller degree than first anticipated by the initial results. Considering this, using hierarchical clustering solely to reduce estimation error is likely to not yield any major benefit for practitioners.

Already in section 5.2, it becomes evident that the NCO portfolios are performing worse or only marginally better than their classical counterparts in terms of volatility, DR, and $SSPW$. Only the HRP and HERC portfolios provide competitive results in comparison to their benchmark portfolio. If there was any doubt, section 5.3 further concludes the prior results, again indicating that when accounting for instability in the hierarchical clusterings, weight concentrations, and portfolio turnover, these portfolios can be considered to underperform in relation to the traditional benchmark portfolios. The results imply that the clustering-based portfolios are not only generally under-performing in comparison with many of the traditional portfolios included; they also generally incur higher portfolio turnover, and depending on the linkage method, higher weight concentrations.

From a practitioner's view, the caveats of these portfolio optimization methods should be carefully considered before applied in a live portfolio at this stage. In particular, the NCO portfolios seem to be dominated across most metrics by their traditional benchmarks. The traditional portfolios consistently provide more robust performance, and should, for this reason, be premiered. However, there seems to be no optimal option across all metrics, and the strategies applied should fit the investment objective first and foremost. As such, when looking for low volatility, even in stressed market conditions, the MV portfolio continues to show the most considerable promise. On the other end, the MD portfolio showed excellent risk-adjusted returns over the researched period, suggesting to be riskier, but also more rewarding in terms of returns, indicated by high a Sharpe and Sortino ratio. In between these two extremes, the ERC portfolio produces somewhat worse risk-based performance, but also implied excellent weight diversification and low portfolio turnover. From this perspective, the best portfolio to implement depends on the characteristics of the practitioner's goals, including but not limited to; risk tolerance, diversification requirements, and transaction costs.

6.5 Further Research

The interpretation of the empirical results of this thesis is that a large part of the cluster formation instability can be attributed to the misspecification of the correlation matrix. Considering this, it is not clear how robust the correlation clustering is to estimation noise. Therefore, further research could help to identify potential covariance pre-processing methods to reduce noise, and thus increase the validity of clusterings used over time. It would mainly be interesting to further look into variants where the market mode could be reduced even more, potentially producing more distinguished clusterings, as well as more diagonalized results in the between-cluster allocation step used in the NCO method.

Much research is yet to be made regarding the optimal clustering algorithm for financial correlation clustering. Hierarchical clustering is one of many alternatives to be applied, and although it is popular in finance, several other methods could prove more fruitful. Further research on the intertemporal cluster stability produced by different methods could help determining the most suitable method, as this instability is assessed to be one of the most significant shortcomings of hierarchical clustering found in this thesis. In addition, alternative data sources for the generation of clusterings could prove less unstable. As clustering methods are agnostic to the distances used, other sources of financial data could be utilized, potentially producing more robust clusters over time. Furthermore, additional research into the impact of different rebalancing schemes and optimal rebalancing for hierarchical clustering-based portfolio methods is highly relevant. High portfolio turnover is one of the main drawbacks seen across all aforementioned allocation methods, and reducing it could make them more competitive and applicable in practice.

With the findings of this thesis, it is hard to provide results that generalize over all different market regimes. Therefore, further research is warranted to further investigating how the results and potential benefits from hierarchical clustering change under different market periods. With the COVID-19 crisis, the world is again reminded that markets are highly unpredictable and that before implementing a strategy in practice, careful considerations should be taken to consider potential pitfalls that can occur during stressed markets. As such, a topic of further research could be to investigate how the cluster quality changes during different market modes and to assess how robust these methods are in finding true relationships between assets and clusters.

7 Conclusion

The objective of this thesis is to provide further evidence and findings over how hierarchical clustering portfolios perform from a risk-based perspective in comparison to each other and their equivalent non-hierarchical benchmarks. To answer the research question, several sub-questions were posed. In the concluding part of this thesis, the overarching conclusion is presented, and each sub-question is answered separately.

In conclusion, this study finds that hierarchical clustering-based portfolio optimization techniques underperform the traditional method benchmarks in most risk-based dimensions. Nested Clustered Optimization shows limited applicability, while Hierarchical Risk Parity and Hierarchical Equal Risk Contribution provide competitive results in comparison to their inverse-variance benchmark. The main drawbacks are assessed to be the intertemporal instability of weight allocations, depicted by high portfolio turnover. However, the overall performance can be said to be fair, and more research is needed to reach more general conclusions. Below, the research questions are answered to conclude the thesis.

Can the Estimation Error of the Traditional Portfolios be Reduced using Nested Clustered Optimization?

The findings from the first Monte Carlo backtest, featured in section 5.1 shows inconclusive results regarding the potential benefits of hierarchical clustering to reduce estimation error through the NCO method. At first, using a simple stylized correlation matrix, the method indicates substantial gains. However, these benefits are found to be diminishing with more realistic financial data, suggesting that the method is non-robust using the implementation as provided in this thesis.

How do Different Linkage Methods Affect the Portfolio Optimization Results?

The empirical findings indicate that the choice of linkage method has material implications on the weight allocation characteristics of the portfolios, especially on the NCO and HERC portfolios. The results further imply that the linkage method has a different impact depending on the optimization objective. Therefore, no general guidelines can be inferred from this thesis. However, in the general case, linkage methods that produce more well-balanced clusters show more promise in providing lower weight concentrations in the portfolio construction. As such, Ward's Method produces the most diversified portfolios in terms of weight concentration, while Single Linkage produces the most dense weight concentrations.

How well do the Portfolios Perform from a Risk-Based Perspective?

The empirical findings indicate that the traditional portfolios are superior in terms of their own optimization objective. From the empirical results, it was found that the different NCO portfolios underperformed the traditional portfolios in terms of their optimization objective and many of the risk-based performance metrics. The results were consistent between the Monte Carlo backtest and the historical backtest, increasing the validity of said findings. Both the HRP and HERC portfolios displayed competitive performance and outperformed their closest benchmark in terms of volatility and in all included risk-based performance metrics. However, the overall best performing portfolios from a risk-based perspective are assessed to be the MV and MD portfolios, indicating high Sharpe and Sortino ratios, while also displaying competitive levels of the Maximum Drawdown, Value-at-Risk (VaR), and Conditional Value-at-Risk (CVaR).

How Concentrated are the Weights of the Different Portfolios?

The diversification in terms of weight concentration measured by $SSPW$ indicated that the hierarchical clustering-based portfolios are flexible, and that the choice of linkage method is the most substantial contributor to the weight concentration. This was illustrated in the Monte Carlo backtest as Ward's Method consistently provided low weight concentrations, regardless of the optimization objective. Most weight diversification was, as expectedly, achieved by the EW portfolio as it has perfect weight diversification by definition. Following this, the ERC, IV, NCO ERC, HRP, and HERC portfolios produced well-diversified portfolios. The highest weight concentration was found in the MV portfolios followed by the MD portfolios.

What Level of Asset Turnover is Incurred by the Different Portfolios?

Previous studies have highlighted the fact that clustering-based portfolios tend to incur higher turnover rates. The empirical findings of this thesis are in agreement with the previous results, providing further evidence that this is one of the downsides in applying these methods. In the historical backtest, the NCO portfolios show higher rates compared to their traditional counterparts. In addition, the HRP and HERC portfolios show a higher turnover rate than the IV portfolio but are still competitive in comparison to the traditional MV and MD portfolios.

References

Ardia, D., Bolliger, G., Boudt, K., & Gagnon-Fleury, J.-P. (2017). The impact of covariance misspecification in risk-based portfolios. *Annals of Operations Research*, *254*(1-2), 1–16.

Arnott, R., Harvey, C. R., & Markowitz, H. (2019). A backtesting protocol in the era of machine learning. *The Journal of Financial Data Science*, *1*(1), 64–74.

Bai, X., Scheinberg, K., & Tutuncu, R. (2016). Least-squares approach to risk parity in portfolio selection. *Quantitative Finance*, *16*(3), 357–376.

Bar-Joseph, Z., Gifford, D. K., & Jaakkola, T. S. (2001). Fast optimal leaf ordering for hierarchical clustering. *Bioinformatics*, *17*(suppl_1), S22–S29.

Bell, E., Bryman, A., & Harley, B. (2018). *Business research methods*. Oxford university press.

Bonanno, G., Caldarelli, G., Lillo, F., & Mantegna, R. N. (2003). Topology of correlation-based minimal spanning trees in real and model markets. *Physical Review E*, *68*(4), 046130.

Bonanno, G., Lillo, F., & Mantegna, R. N. (2001). Levels of complexity in financial markets. *Physica A: Statistical Mechanics and its Applications*, *299*(1-2), 16–27.

Bouchaud, J.-P. & Potters, M. (2009). Financial applications of random matrix theory: a short review. *arXiv preprint arXiv:0910.1205*.

Braga, M. D. (2015). *Risk-based approaches to asset allocation: concepts and practical applications*. Springer.

Bun, J., Bouchaud, J.-P., & Potters, M. (2017). Cleaning large correlation matrices: tools from random matrix theory. *Physics Reports*, *666*, 1–109.

Chekhlov, A., Uryasev, S., & Zabarankin, M. (2005). Drawdown measure in portfolio optimization. *International Journal of Theoretical and Applied Finance*, *8*(01), 13–58.

Choueifaty, Y. (2006). Methods and systems for providing an anti-benchmark portfolio. USPTO 60/816, 276.

Choueifaty, Y. & Coignard, Y. (2008, October). Toward maximum diversification. *Journal of Portfolio Management - J PORTFOLIO MANAGE*, *35*, 40–51. doi:10.3905/JPM.2008.35.1.40

Choueifaty, Y., Froidure, T., & Reynier, J. (2013, March). Properties of the most diversified portfolio. *The Journal of Investment Strategies*, *2*, 49–70. doi:10.21314/JOIS.2013.033

Clarke, R. G., De Silva, H., & Thorley, S. (2006). Minimum-variance portfolios in the us equity market. *The Journal of Portfolio Management*, *33*(1), 10–24.

Coqueret, G. & Guida, T. (2020). Training trees on tails with applications to portfolio choice. *Annals of Operations Research*, 1–41.

Credit Suisse. (2020). Summary edition credit suisse global investment returns yearbook 2020. https://www.credit-suisse.com/media/assets/corporate/docs/about-us/research/publications/credit-suisse-global-investment-returns-yearbook-2020-summary-edition.pdf.

Demey, P., Maillard, S. [Sébastien], & Roncalli, T. (2010). Risk-based indexation. *Available at SSRN 1582998*.

DeMiguel, V., Garlappi, L., & Uppal, R. (2009). Optimal versus naive diversification: how inefficient is the 1/n portfolio strategy? *The review of Financial studies*, *22*(5), 1915–1953.

Dose, C. & Cincotti, S. (2005). Clustering of financial time series with application to index and enhanced index tracking portfolio. *Physica A: Statistical Mechanics and its Applications*, *355*(1), 145–151.

Eisen, M. B., Spellman, P. T., Brown, P. O., & Botstein, D. (1998). Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Sciences*, *95*(25), 14863–14868.

Föllmer, H. & Schied, A. (2011). *Stochastic finance: an introduction in discrete time*. Walter de Gruyter.

Frobenius, G. (1912). Über matrizen aus nicht negativen elementen.

Goetzmann, W. N. & Kumar, A. (2008). Equity portfolio diversification. *Review of Finance*, *12*(3), 433–463.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., … Bengio, Y. (2014). Generative adversarial nets. In *Advances in neural information processing systems* (pp. 2672–2680).

Gower, J. C. (1966). Some distance properties of latent root and vector methods used in multivariate analysis. *Biometrika*, *53*(3-4), 325–338.

Green, R. C. & Hollifield, B. (1992). When will mean-variance efficient portfolios be well diversified? *The Journal of Finance*, *47*(5), 1785–1809.

Greenwood, R. & Shleifer, A. (2014). Expectations of returns and expected returns. *The Review of Financial Studies*, *27*(3), 714–746.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media.

Higham, N. J. (2002). Computing the nearest correlation matrix—a problem from finance. *IMA journal of Numerical Analysis*, *22*(3), 329–343.

Ilmanen, A. & Villalon, D. (2012). Alpha beyond expected returns.

Jain, P. & Jain, S. (2019). Can machine learning-based portfolios outperform traditional risk-based portfolios? the need to account for covariance misspecification. *Risks*, *7*(3), 74.

Kim, H., Kim, I., Lee, Y., & Kahng, B. (2002). Scale-free network in stock markets. *Journal-Korean Physical Society*, *40*, 1105–1108.

Kraskov, A., Stögbauer, H., & Grassberger, P. (2004). Estimating mutual information. *Physical review E*, *69*(6), 066138.

Kwan, C. (2010, July). The requirement of a positive definite covariance matrix of security returns for mean-variance portfolio analysis: a pedagogic illustration. *Spreadsheets in Education (eJSiE)*, *4*.

Laloux, L., Cizeau, P., Potters, M., & Bouchaud, J.-P. (2000). Random matrix theory and financial correlations. *International Journal of Theoretical and Applied Finance*, *3*(03), 391–397.

Ledoit, O. & Wolf, M. (2000). A well conditioned estimator for large dimensional covariance matrices.

Ledoit, O. & Wolf, M. (2004). Honey, i shrunk the sample covariance matrix. *The Journal of Portfolio Management*, *30*(4), 110–119.

Lo, A. (2003, February). The statistics of sharpe ratios. *Financial Analysts Journal*, *58*. doi:10.2469/faj.v58.n4.2453

López de Prado, M. (2016). Building diversified portfolios that outperform out of sample. *The Journal of Portfolio Management*, *42*(4), 59–69.

López de Prado, M. (2019a). A robust estimator of the efficient frontier. *Available at SSRN 3469961.*

López de Prado, M. (2019b). Tactical investment algorithms. *Available at SSRN 3459866.*

López de Prado, M. (2020). *Machine learning for asset managers*. Elements in Quantitative Finance. Cambridge University Press. doi:10.1017/9781108883658

Maillard, S. [Sebastien], Teiletche, J., & Roncalli, T. (2008, September). On the properties of equally-weighted risk contributions portfolios. doi:10.2139/ssrn.1271972

Mantegna, R. N. (1999). Hierarchical structure in financial markets. *The European Physical Journal B-Condensed Matter and Complex Systems*, *11*(1), 193–197.

Marčenko, V. A. & Pastur, L. A. (1967). Distribution of eigenvalues for some sets of random matrices. *Mathematics of the USSR-Sbornik*, *1*(4), 457.

Markowitz. (1952). Portfolio selection. *The Journal of Finance*, *7*(1), 77–91. Retrieved from http://www.jstor.org/stable/2975974

Markowitz. (1959). *Portfolio selection: efficient diversification of investments.* Yale University Press. Retrieved from http://www.jstor.org/stable/j.ctt1bh4c8h

Marti, G. (2019). Corrgan: sampling realistic financial correlation matrices using generative adversarial networks. *arXiv preprint arXiv:1910.09504*.

Michaud, R. O. (1989). The markowitz optimization enigma: is 'optimized'optimal? *Financial Analysts Journal*, *45*(1), 31–42.

Müllner, D. (2011). Modern hierarchical, agglomerative clustering algorithms. *arXiv preprint arXiv:1109.2378*.

Munk, C. (2018). *Financial markets and investments.*

Sp u.s. indices methodology. (2020), 6–9.

Papenbrock, J. (2011). *Asset clusters and asset networks in financial risk management and portfolio optimization* (Doctoral dissertation, Karlsruher Institut für Technologie (KIT)).

Pasini, G. (2017). Principal component analysis for stock portfolio management. *International Journal of Pure and Applied Mathematics*, *115*(1), 153–167.

Perron, O. (1907). Zur theorie der matrices. *Mathematische Annalen*, *64*(2), 248–263.

Qian, E. (2005). Risk parity portfolios: efficient portfolios through true diversification. *PanAgora Asset Management.*

Raffinot, T. (2017). Hierarchical clustering-based asset allocation. *The Journal of Portfolio Management*, *44*(2), 89–99.

Raffinot, T. (2018). The hierarchical equal risk contribution portfolio. *Available at SSRN 3237540.*

Ross, S. A. (1976). The arbitrage theory of capital asset pricing. *Journal of Economic Theory*, *13*(3), 341–360.

Sharpe, W. F. (1964). Capital asset prices: a theory of market equilibrium under conditions of risk. *The journal of finance*, *19*(3), 425–442.

Sortino, F. A. & Price, L. N. (1994). Performance measurement in a downside risk framework. *the Journal of Investing*, *3*(3), 59–64.

Spinu, F. (2013). An algorithm for computing risk parity weights. *Available at SSRN 2297383.*

Thompson, S. (2013). The stylised facts of stock price movements. *The New Zealand Review of Economics and Finance*, *1.*

Thomson Reuters Eikon. (2020). S&P 500 Total Return Index. Data retrieved from Yahoo Finance on 2020-05-02. Can be accessed at: https://finance.yahoo.com/quote/5ESP500TR/history?p=5ESP500TR.

Tibshirani, R., Walther, G., & Hastie, T. (2001). Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *63*(2), 411–423.

Tokat, Y. & Wicas, N. W. (2007). Portfolio rebalancing in theory and practice. *The Journal of Investing*, *16*(2), 52–59.

van der Meer, R., Sortino, F., & Plantinga, A. (2001). The impact of downside risk on risk-adjusted performance of mutual funds in the euronext markets. *Available at SSRN 277352.*

Vandewalle, N., Brisbois, F., Tordoir, X., et al. (2001). Non-random topology of stock markets. *Quantitative Finance*, *1*(3), 372–374.

Ward Jr, J. H. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, *58*(301), 236–244.

Westfall, P. H. (2014). Kurtosis as peakedness, 1905–2014. rip. *The American Statistician*, *68*(3), 191–195.

Wheeler, D. J. (2011). Problems with skewness and kurtosis, part one. *Retrieved from Quality Digest Web site: www. qualitydigest. com.*

Yue, S., Wang, X., & Wei, M. (2008). Application of two-order difference to gap statistic. *Transactions of Tianjin University*, *14*(3), 217–221.

Zakamulin, V. (2015). A test of covariance-matrix forecasting methods. *The Journal of Portfolio Management*, *41*(3), 97–108.

Zhang, J. & Maringer, D. (2009). Improving sharpe ratios and stability of portfolios by using a clustering technique. In *Proceedings of the world congress on engineering* (Vol. 1).

Appendix

A1 S&P General Information

1. Files 10-K annual reports
2. The U.S. portion of fixed assets and revenues constitutes a plurality of the total, but need not exceed 50 percent. When these factors are in conflict, fixed assets determine plurality. Revenue determines plurality when there is incomplete asset information. Geographic information for revenue and fixed asset allocations are determined by the company as reported in its annual filings.
3. The primary listing must be on an eligible U.S. exchange.

The following list of exchanges are considered eligible primary listings:

1. New York Stock Exchange (NYSE)
2. New York Stock Exchange - Arca
3. New York Stock Exchange - American
4. NASDAQ Global Select Market
5. NASDAQ Select Market
6. NASDAQ Capital Market
7. CBOE BZX
8. CBOE BYX
9. CBOE EDGA
10. CBOE EDGX

Source: “SP U.S. Indices Methodology” (2020)

A2 Stylized Facts of Correlation Matrix

A3 Code – Portfolio Implementations

```
from datetime import datetime
import time
from tqdm import tqdm
```

```python
import numpy as np
import pandas as pd
import math
import random

import matplotlib as mpl
from matplotlib import pyplot as plt
import matplotlib.dates as mdates
import matplotlib.ticker as mtick
import seaborn as sns

import scipy.cluster.hierarchy as hierarchy
from scipy.spatial.distance import pdist
from pypfopt.risk_models import risk_matrix
from scipy.linalg import block_diag
from sklearn.utils import check_random_state
import cvxpy as cp
\end{}

\subsubsection{Minimum Variance Portfolio}

\begin{minted}[breaklines]{python}
class MinimumVariance:
    def __init__(self):
        self.weights = None

    def allocate(self, cov, long_only=True):
        cov = pd.DataFrame(cov)
        n = len(cov) # number of assets
        w = cp.Variable(n) # initialize weights
        risk = cp.quad_form(w, cov) # portfolio variance
        prob = cp.Problem(cp.Minimize(0.5 * risk), [cp.sum(w) == 1, w >= 0])
        prob.solve() # solve problem
        self.weights = pd.Series(w.value, index=cov.index)
        return self.weights
```

A3.1 Equal-Risk Contribution Portfolio

```
class InverseVariance:
    def __init__(self):
        self.weights = None

    def allocate(self, cov):
        cov = pd.DataFrame(cov)
        inv_diag = 1 / np.diag(cov.values)
        weights = inv_diag * (1 / np.sum(inv_diag))
        self.weights = pd.Series(weights, index=cov.columns)
        return self.weights

class InverseVolatility:
    def __init__(self):
        self.weights = None

    def allocate(self, cov):
        cov = pd.DataFrame(cov)
        inv_diag = 1 / np.sqrt(np.diag(cov.values))
        weights = inv_diag * (1 / np.sum(inv_diag))
        self.weights = pd.Series(weights, index=cov.columns)
        return self.weights

class EqualRiskContribution:
    def __init__(self):
        self.weights = None

    def allocate(self, cov):
        cov = pd.DataFrame(cov)
        n = len(cov) # number of assets
        w = cp.Variable(n) # initilize weights
        risk = cp.quad_form(w, cov) # portfolio variance
        log_w = cp.sum((1/n)cp.log(w)) # log weights
        prob = cp.Problem(cp.Minimize(0.5 * risk - log_w), [w >= 0.0001])
        prob.solve() # optimize weights
        self.weights = pd.Series(w.value/np.sum(w.value), index=cov.index)
        return self.weights
```

A3.2 Maximum Diversification

```
class MaximumDiversification:
    def __init__(self):
        self.weights = None

    def allocate(self, cov, long_only=True):
        cov = pd.DataFrame(cov)
        n = len(cov) # number of assets
        w = cp.Variable(n) # initialize weights
        vol = cp.sqrt(cp.diag(cov))
        w_vol = cp.multiply(w, vol)
        risk = cp.quad_form(w, cov) # portfolio volatility
        prob = cp.Problem(cp.Minimize(0.5 * risk), [cp.sum(w_vol) == 1, w >= 0])
        prob.solve() # solve problem
        self.weights = pd.Series(w.value/np.sum(w.value), index=cov.index)
        return self.weights
```

A3.3 Hierarchical Risk Parity

```
class HierarchicalRiskParity:
    def __init__(self):
        self.weights = None
        self.asset_order = None
        self.clusters = None
        self.corr = None
        self.corr_sorted = None
```

```
# get inverse-variance/volatility weights
def inv_var(self, cov, w_method):
    cov = pd.DataFrame(cov)
    if w_method == 'vol':
        inv_diag = 1 / np.sqrt(np.diag(cov.values))
    else:
        inv_diag = 1 / np.diag(cov.values)
    weight = inv_diag * (1 / np.sum(inv_diag))
    weight = weight.reshape(-1,1)
    return weight

# create hierarchical clustering
def hierarchical_clustering(self, corr, linkage='single', leaf_order=False):
    dist = np.sqrt((1 - corr).round(5) / 2) # calculate distances
    p_dist = pdist(dist) # transform to distance matrix
    clusters = hierarchy.linkage(p_dist, method=linkage, optimal_ordering=leaf_order) # c
    return clusters

# sort clustered items by distance
def seriation(self, clusters):
    return hierarchy.leaves_list(clusters)

# compute HRP weight allocation through recursive bisection
def recursive_bisection(self, cov, sort_order, w_method):
    weight = pd.Series(1, index=sort_order) # set initial weights to 1
    items = [sort_order]

    while len(items) > 0: # loop while weights is under 100%
        items = [i[j:k]
                 for i in items
                 for j, k in ((0, len(i) // 2), (len(i) // 2, len(i))) # get cluster indi
                 if len(i) > 1
                ]

        # allocate weight to left and right cluster
```

```
            for i in range(0, len(items), 2):
                left_cluster = items[i]
                right_cluster = items[i + 1]

                # Left cluster
                left_cov = cov.iloc[left_cluster, left_cluster]
                left_weight = self.inv_var(left_cov, w_method)
                left_var = np.dot(np.dot(left_weight.T, left_cov), left_weight)[0,0]

                # Right cluster
                right_cov = cov.iloc[right_cluster, right_cluster]
                right_weight = self.inv_var(right_cov, w_method)
                right_var = np.dot(np.dot(right_weight.T, right_cov), right_weight)[0,0]

                # Allocate weight to clusters
                alpha = 1 - left_var / (left_var + right_var)
                weight[left_cluster] *= alpha # weight 1
                weight[right_cluster] *= 1 - alpha # weight 2

        self.weights = weight
        self.weights.index = self.asset_order

    # Transform covariance matrix to correlation matrix
    def cov2corr(self, cov):
        std = np.sqrt(np.diag(cov))
        corr = cov / np.outer(std, std)
        corr[corr < -1], corr[corr > 1] = -1, 1 # numerical error
        return corr

    # Allocate weights
    def allocate(self, cov, linkage='single', leaf_order=True, w_method='var'):

        # Correlation matrix from covariance matrix
        self.corr = pd.DataFrame(self.cov2corr(cov))
        # Step-1: Tree clustering
        self.clusters = self.hierarchical_clustering(self.corr, linkage, leaf_order)

        # Step-2: Seriation (Quasi-Diagnalization)
        self.sort_order = self.seriation(self.clusters)

        asset_order = list(cov.columns)
        asset_order[:] = [asset_order[i] for i in self.sort_order]
        self.asset_order = asset_order
        self.corr_sorted = self.corr.reindex(index=self.asset_order, columns=self.asset_order

        # Step-3: Recursive bi-section
        self.recursive_bisection(cov, self.sort_order, w_method)
        return self.weights
```

A3.4 Hierarchical Equal Risk Contribution

```
class HierarchicalEqualRiskContribution:
    def __init__(self):
        self.weights = None
        self.asset_order = None
        self.clusters = None
        self.corr = None
        self.corr_sorted = None

    # get inverse-variance/volatility weights
    def inv_var(self, cov, w_method='var'):
        cov = pd.DataFrame(cov)
        if w_method == 'vol':
            inv_diag = 1 / np.sqrt(np.diag(cov.values))
        else:
            inv_diag = 1 / np.diag(cov.values)
        weight = inv_diag * (1 / np.sum(inv_diag))
        weight = weight.reshape(-1,1)
        return weight
```

```
# create hierarchical clustering
def hierarchical_clustering(self, corr, linkage, max_k):
    # hierarchcial clustering
    dist = np.sqrt((1 - corr).round(5) / 2)
    dist = pd.DataFrame(dist, columns=corr.columns, index=corr.index)
    p_dist = pdist(dist)
    clustering = hierarchy.linkage(p_dist, method=linkage)

    # cluster levels over from 1 to N-1 clusters
    cluster_lvls = pd.DataFrame(hierarchy.cut_tree(clustering),
    ↪  index=corr.columns)
    num_k = cluster_lvls.columns # save column with number of clusters
    cluster_lvls = cluster_lvls.iloc[:, ::-1] # reverse order to start with
    ↪  1 cluster
    cluster_lvls.columns = num_k # set columns to number of cluster
    W_list = []

    # get within-cluster dissimilarity for each k
    for k in range(min(len(cluster_lvls.columns), max_k)):
        level = cluster_lvls.iloc[:,k] # get k clusters
        D_list = [] # within-cluster distance list
        for i in range(np.max(level.unique())+1):
            cluster = level.loc[level == i] #
            cluster_dist = dist.loc[cluster.index, cluster.index] # get
            ↪  distance
            cluster_pdist = pdist(cluster_dist) # flatten and transform to
            ↪  squared euclidean
            D = np.nan_to_num(cluster_pdist.std())
            D_list.append(D) # append to list
        W_k = np.sum(D_list)
        W_list.append(W_k)

    W_list = pd.Series(W_list)
```

```
        n = corr.shape[0]
        limit_k = int(min(max_k, np.sqrt(n)))
        gaps = W_list.shift(-2) + W_list - 2*W_list.shift(-1)
        gaps = gaps[0:limit_k]
        k = gaps.idxmax() + 2

        return clustering, k

    # sort clustered items by distance
    def seriation(self, clusters):
        return hierarchy.leaves_list(clusters)

    # compute HRP weight allocation through cluster-based bisection
    def hierarchical_recursive_bisection(self, cov, linkage, k, w_method):

        # Transform linkage to tree and reverse order
        root, nodes = hierarchy.to_tree(linkage, rd=True)
        nodes = nodes[::-1]
        items = pd.Series(data=hierarchy.leaves_list(linkage)) # Get list of
         ↪  assets
        weight = pd.Series(1, index=cov.index) # Set initial weights to 1

        # Loop through k clusters
        for i in nodes[:k-1]:
            if i.is_leaf() == False: # skip leaf-nodes
                left = i.get_left().pre_order(lambda i: i.id) # get left
                 ↪  cluster
                right = i.get_right().pre_order(lambda i: i.id) # get right
                 ↪  cluster

                # Left cluster
                left_cov = cov.iloc[left, left]
                left_weight = self.inv_var(left_cov, w_method)
                left_var = np.dot(np.dot(left_weight.T, left_cov),
                 ↪  left_weight)[0,0]

                # Right cluster
                right_cov = cov.iloc[right, right]
                right_weight = self.inv_var(right_cov, w_method)
                right_var = np.dot(np.dot(right_weight.T, right_cov),
                 ↪  right_weight)[0,0]

                # Allocate weight to clusters
                alpha = 1 - left_var / (left_var + right_var)
                weight[left] *= alpha # weight 1
                weight[right] *= 1 - alpha # weight 2

        # Get constituents of k clusters
        clustered_assets = pd.Series(hierarchy.cut_tree(linkage,
         ↪  n_clusters=k).flatten(), index=cov.index)
        w = pd.DataFrame(np.ones(len(cov.index)), index=cov.index)

        # Multiply within-cluster weight with inter-cluster weight
        for i in range(k):
            cluster = clustered_assets.loc[clustered_assets == i]
            cluster_cov = cov.loc[cluster.index, cluster.index]
            cluster_weights = pd.Series(self.inv_var(cluster_cov,
             ↪  w_method).flatten(), index=cluster_cov.index)
            weight.loc[cluster_weights.index] *= cluster_weights
        return weight

    # Transform covariance matrix to correlation matrix
    def cov2corr(self, cov):
        std = np.sqrt(np.diag(cov))
        corr = cov / np.outer(std, std)
        corr[corr < -1], corr[corr > 1] = -1, 1 #numerical error
        return corr
    # ----------- Output Methods ----------- #

    # Allocate weights
    def allocate(self, cov, linkage='ward', max_k=10, w_method='var'):

        # Correlation matrix from covariance matrix
        self.corr = self.cov2corr(cov)

        # Step-1: Tree clustering
        self.clusters, self.k,  = self.hierarchical_clustering(self.corr,
        ↪  linkage, max_k)

        # Step-2: Seriation (Quasi-Diagnalization)
        self.sort_order = self.seriation(self.clusters)

        asset_order = list(cov.columns)
        asset_order[:] = [asset_order[i] for i in self.sort_order]
        self.asset_order = asset_order
        self.corr_sorted = self.corr.reindex(index=self.asset_order,
        ↪  columns=self.asset_order)

        # Step-3: Recursive Bisection
        self.weights = self.hierarchical_recursive_bisection(cov, self.clusters,
        ↪  self.k, w_method)
        return self.weights
```

A3.5 Nested Clustered Optimization

```
class NestedClusteredOptimization:
    def __init__(self):
        self.corr = None
        self.corr_sorted = None
        self.clusters = None
        self.k = None
        self.weights = None

    # Minimum-Variance Allocation
    def opt_w(self, cov, w_method, long_only=True):
        cov = pd.DataFrame(cov)
        n = len(cov) # number of assets

        # Minimum Variance
        if w_method == 'mv':
            w = cp.Variable(n) # initialize weights
            risk = cp.quad_form(w, cov) # portfolio variance
            prob = cp.Problem(cp.Minimize(0.5 * risk), [cp.sum(w) == 1, w >= 0])
            prob.solve() # solve problem
            weight = w.value

        # Equal Risk Contribution
        if w_method == 'md':
            w = cp.Variable(n) # initialize weights
            vol = cp.sqrt(cp.diag(cov))
            w_vol = cp.multiply(w, vol)
            risk = cp.quad_form(w, cov) # portfolio volatility
            prob = cp.Problem(cp.Minimize(0.5 * risk), [cp.sum(w_vol) == 1, w >=
             ↪  0])
            prob.solve() # solve problem
            weight = w.value/np.sum(w.value)

        # Maximum Diversification
        if w_method == 'erc':
            w = cp.Variable(n) # initilize weights
            risk = cp.quad_form(w, cov) # portfolio variance
            log_w = cp.sum((1/n)*cp.log(w)) # log weights
            prob = cp.Problem(cp.Minimize(0.5 * risk - log_w), [w >= 0]) #
             ↪  defined problem
            prob.solve() # optimize weights
            weight = w.value/np.sum(w.value)

        weight = pd.Series(weight, index=cov.index)
        return weight

    # create hierarchical clustering
    def hierarchical_clustering(self, corr, linkage, max_k):
        # hierarchcial clustering
        dist = np.sqrt((1 - corr).round(5) / 2)
        dist = pd.DataFrame(dist, columns=corr.columns, index=corr.index)
        p_dist = pdist(dist)
        clustering = hierarchy.linkage(p_dist, method=linkage)

        # cluster levels over from 1 to N-1 clusters
        cluster_lvls = pd.DataFrame(hierarchy.cut_tree(clustering),
         ↪  index=corr.columns)
        num_k = cluster_lvls.columns # save column with number of clusters
        cluster_lvls = cluster_lvls.iloc[:, ::-1] # reverse order to start with
         ↪  1 cluster
        cluster_lvls.columns = num_k # set columns to number of cluster
        W_list = []

        # get within-cluster dissimilarity for each k
        for k in range(min(len(cluster_lvls.columns), max_k)):
            level = cluster_lvls.iloc[:,k] # get k clusters
            D_list = [] # within-cluster distance list
            for i in range(np.max(level.unique())+1):
                cluster = level.loc[level == i]
                cluster_dist = dist.loc[cluster.index, cluster.index] # get
                 ↪  distance
                cluster_pdist = pdist(cluster_dist) # flatten and transform to
                 ↪  squared euclidean
                D = np.nan_to_num(cluster_pdist.std())
                D_list.append(D) # append to list
            W_k = np.sum(D_list)
            W_list.append(W_k)

        W_list = pd.Series(W_list)
        n = corr.shape[0]
        limit_k = int(min(max_k, np.sqrt(n)))
        gaps = W_list.shift(-2) + W_list - 2*W_list.shift(-1)
        gaps = gaps[0:limit_k]
        k = gaps.idxmax() + 2

        return clustering, k

    # compute intra-cluster weights
    def intra_weights(self, cov, clusters, k, w_method):
        clusters = hierarchy.cut_tree(clusters, n_clusters=k)
        clusters = pd.DataFrame(clusters)
        clusters.columns = ['cluster']
        clusters.index = cov.index

        # get covariance matrices for each cluster
        intra_weights = pd.DataFrame(index=clusters.index)
        for i in range(k):
            cluster = clusters.loc[clusters.cluster == i]
            cluster_cov = cov.loc[cluster.index, cluster.index]
            weights = pd.Series(self.opt_w(cluster_cov, w_method),
            ↪   index=cluster_cov.index)
            intra_weights[i] = weights

        intra_weights = intra_weights.fillna(0)
        return intra_weights

    def inter_weights(self, cov, intra_weights, w_method):
        # inter-cluster covariance matrix
        tot_cov = intra_weights.T.dot(np.dot(cov, intra_weights))
        # inter-cluster weights
        inter_weights = pd.Series(self.opt_w(tot_cov, w_method),
         ↪  index=intra_weights.columns)

        # determine the weight on each cluster by multiplying the intra-cluster
         ↪  weight with the inter-cluster weight
        weights = intra_weights.mul(inter_weights,
         ↪  axis=1).sum(axis=1).sort_index()
        return weights

    # Transform covariance matrix to correlation matrix
    # Covariance to Correlation transformation
    def cov2corr(self, cov):
        std = np.sqrt(np.diag(cov))
        corr = cov / np.outer(std, std)
        corr[corr < -1], corr[corr > 1] = -1, 1 #numerical error
        return corr

    # ----------- Output Methods ----------- #

    # Allocate weights
    def allocate(self, cov, linkage='ward', max_k=10, w_method='mv'):

        # Correlation matrix from covariance matrix
        self.corr = self.cov2corr(cov)

        # Step-1: Tree clustering
        self.clusters, self.k,  = self.hierarchical_clustering(self.corr,
         ↪  linkage, max_k)

        # Step-2: Determine intra-cluster weights
        intra_weights = self.intra_weights(cov, self.clusters, self.k, w_method)
```

```
        # Step-3: Determine inter-cluster weights and multiply with
        ↪   intra-cluster weights
        self.weights = self.inter_weights(cov, intra_weights, w_method)
        return self.weights
```

A4 Code – Backtesting

A4.1 Monte Carlo Backtest: Estimation Error

```
# Correlation and Covariance Transformation
def corr2cov(corr,std):
    cov = corr * np.outer(std, std)
    return cov

def cov2corr(cov):
        std = np.sqrt(np.diag(cov))
        corr = cov / np.outer(std, std)
        corr[corr < -1], corr[corr > 1] = -1, 1 #numerical error
        return corr

# Create block-diagonal correlation matrix with variable sized clusters

def getCovSub(nObs, nCols, sigma, random_state=None):
    # Sub correl matrix
    rng = check_random_state(random_state)
    if nCols == 1:
        return np.ones((1,1))
    ar0 = rng.normal(size=(nObs,1))
    ar0 = np.repeat(ar0, nCols, axis=1)
    ar0 += rng.normal(scale=sigma, size=ar0.shape)
    ar0 = np.cov(ar0, rowvar=False)
    return ar0
```

```
def getRndBlockCov(nCols, nBlocks, minBlockSize=1, sigma=1.0,
↪ random_state=None):
    # Generate a block random correlation matrix
    rng = check_random_state(random_state)
    parts = rng.choice(range(1, nCols-(minBlockSize-1)*nBlocks), nBlocks-1,
    ↪ replace=False)
    parts.sort()
    parts = np.append(parts, nCols-(minBlockSize-1)*nBlocks)
    parts = np.append(parts[0],np.diff(parts)) - 1 + minBlockSize
    cov = None
    for nCols_ in parts:
        cov_ = getCovSub(int(max(nCols_*(nCols_+1)/2.,100)), nCols_, sigma,
        ↪ random_state=rng)
        if cov is None:
            cov = cov_.copy()
        else:
            cov = block_diag(cov,cov_)
    return cov

def randomBlockCorr(nCols, nBlocks, random_state=None, minBlockSize=1):
    # Form block corr
    rng = check_random_state(random_state)
    cov0 = getRndBlockCov(nCols, nBlocks, minBlockSize=minBlockSize, sigma=0.5,
    ↪ random_state=rng)
    cov1 = getRndBlockCov(nCols, 1, minBlockSize=minBlockSize, sigma=1.0,
    ↪ random_state=rng) # add noise
    cov0 += cov1
    corr0 = cov2corr(cov0)
    corr0 = pd.DataFrame(corr0)
    return corr0

# Create block-diagonal correlation matrix with fixed cluster size
def formBlockMatrix(nBlocks, covSize, bCorr):
    bSize = int(covSize / nBlocks)
    block = np.ones((bSize, bSize)) * bCorr
    block[range(bSize), range(bSize)] = 1
    corr = block_diag(*([block]*nBlocks))
    return corr

# Take block matrix as input
def formTrueMatrix(corr):
    corr = pd.DataFrame(corr)
    cols = corr.columns.tolist()
    np.random.shuffle(cols)
    corr = corr[cols].loc[cols].copy(deep=True)
    std = np.random.uniform(0.05, 0.2, corr.shape[0])
    cov = corr2cov(corr, std)
    return cov

# Simulate Empirical Covariance Matrix
def simCov(cov0, nObs, shrink=False):
    x = np.random.multivariate_normal(np.zeros((cov0.shape[0],)), cov0,
    ↪  size=nObs)
    x = pd.DataFrame(x)
    if shrink:
        cov1 = risk_matrix(x, method='ledoit_wolf_constant_correlation',
        ↪  returns_data=True)
    else:
        cov1 = np.cov(x, rowvar=0)
    return cov1

# set seed for replicability
np.random.seed(0)

# Variables for True Covariance Matrix
nBlocks = 10
covSize = 100
bCorr = 0.5
```

```
nCols = 100
minBlockSize = 5
random = False # determines the correlation matrix used

# Variables for Monte Carlo
nObs = 504
nSims = 200

# Minimum Variance Portfolios

# Initiate DataFrame for Statistics
stats = pd.DataFrame()

# Get True Weights
w0 = pd.DataFrame(columns=range(nCols), index=range(nSims), dtype=float)

# Initiate DataFrames for estimated weights
w1 = w0.copy(deep=True)
w1_single = w0.copy(deep=True)
w1_average = w0.copy(deep=True)
w1_ward = w0.copy(deep=True)

true_k = []
k_single = []
k_average = []
k_ward = []

# Run for both Raw and with Shrinkage
for shrink in [False, True]:

    # Run Monte-Carlo simulations
    for i in tqdm(range(nSims)):

        nBlocks = np.random.randint(5, 10)
```

```
 # Get true covariance matrix and set true weights
 if random=True:
             corr = randomBlockCorr(nCols, nBlocks,
              ↪  minBlockSize=minBlockSize)
 else:
      corr = formBlockMatrix(nBlocks, covSize, bCorr)

cov0 = formTrueMatrix(corr)



 w0.loc[i] = mv.allocate(cov0)

 # Get sample covariance matrix and set weights
 cov1 = simCov(cov0, nObs, shrink=shrink)
 #q = nObs/cov0.shape[0]
 #cov1 = pd.DataFrame(deNoiseCov(cov1, q, bWidth=.01))
 w1.loc[i] = mv.allocate(cov1)

 # Single linkage
 w1_single.loc[i] = nco.allocate(cov1, linkage='single', max_k=15,
  ↪  w_method='mv')
 k_single.append(nco.k)

 # Average linkage
 w1_average.loc[i] = nco.allocate(cov1, linkage='average', max_k=15,
  ↪  w_method='mv')
 k_average.append(nco.k)

 # Ward's Method
 w1_ward.loc[i] = nco.allocate(cov1, linkage='ward', max_k=15,
  ↪  w_method='mv')
 k_ward.append(nco.k)
```

```
        # Save estimated number of clusters (k)
        true_k.append(nBlocks)

    row = []

    # Calculate Root-Mean-Square Errors between weights
    row.append(np.mean((w1 - w0).values.flatten() ** 2 ) ** 0.5)
    row.append(np.mean((w1_single - w0).values.flatten() ** 2) ** 0.5)
    row.append(np.mean((w1_average - w0).values.flatten() ** 2) ** 0.5)
    row.append(np.mean((w1_ward - w0).values.flatten() ** 2) ** 0.5)

    row = pd.Series(row)
    stats = stats.append(row, ignore_index=True)

stats.columns = ['MV', 'Single', 'Average', 'Ward']
stats.index = ['Raw','Shrunk']

# Equal Risk Contribution Portfolios

# Initiate DataFrame for Statistics
stats = pd.DataFrame()

# Get True Weights
w0 = pd.DataFrame(columns=range(nCols), index=range(nSims), dtype=float)

# Initiate DataFrames for estimated weights
w1 = w0.copy(deep=True)
w1_single = w0.copy(deep=True)
w1_average = w0.copy(deep=True)
w1_ward = w0.copy(deep=True)

true_k = []
k_single = []
k_average = []
```

```
k_ward = []

# Run for both Raw and with Shrinkage
for shrink in [False, True]:

    # Run Monte-Carlo simulations
    for i in tqdm(range(nSims)):

        nBlocks = np.random.randint(5, 10)

        # Get true covariance matrix and set true weights
        if random=True:
                    corr = randomBlockCorr(nCols, nBlocks,
                    ↪  minBlockSize=minBlockSize)
        else:
             corr = formBlockMatrix(nBlocks, covSize, bCorr)

       cov0 = formTrueMatrix(corr)
       w0.loc[i] = erc.allocate(cov0)

        # Get sample covariance matrix and set weights
        cov1 = simCov(cov0, nObs, shrink=shrink)
        #q = nObs/cov0.shape[0]
        #cov1 = pd.DataFrame(deNoiseCov(cov1, q, bWidth=.01))
        w1.loc[i] = erc.allocate(cov1)

        # Single linkage
        w1_single.loc[i] = nco.allocate(cov1, linkage='single', max_k=15,
        ↪  w_method='erc')
        k_single.append(nco.k)

        # Average linkage
        w1_average.loc[i] = nco.allocate(cov1, linkage='average', max_k=15,
        ↪  w_method='erc')
        k_average.append(nco.k)

        # Ward's Method
        w1_ward.loc[i] = nco.allocate(cov1, linkage='ward', max_k=15,
         ↪  w_method='erc')
        k_ward.append(nco.k)

        # Save estimated number of clusters (k)
        true_k.append(nBlocks)

    row = []

    # Calculate Root-Mean-Square Errors between weights
    row.append(np.mean((w1 - w0).values.flatten() ** 2 ) ** 0.5)
    row.append(np.mean((w1_single - w0).values.flatten() ** 2) ** 0.5)
    row.append(np.mean((w1_average - w0).values.flatten() ** 2) ** 0.5)
    row.append(np.mean((w1_ward - w0).values.flatten() ** 2) ** 0.5)

    row = pd.Series(row)
    stats = stats.append(row, ignore_index=True)

stats.columns = ['MV', 'Single', 'Average', 'Ward']
stats.index = ['Raw','Shrunk']

# ---------------------------------
# Maximum Diversification
# ---------------------------------
# Initiate DataFrame for Statistics
stats = pd.DataFrame()

# Get True Weights
w0 = pd.DataFrame(columns=range(nCols), index=range(nSims), dtype=float)

# Initiate DataFrames for estimated weights
```

```
w1 = w0.copy(deep=True)
w1_single = w0.copy(deep=True)
w1_average = w0.copy(deep=True)
w1_ward = w0.copy(deep=True)

true_k = []
k_single = []
k_average = []
k_ward = []

# Run for both Raw and with Shrinkage
for shrink in [False, True]:

    # Run Monte-Carlo simulations
    for i in tqdm(range(nSims)):

        nBlocks = np.random.randint(5, 10)

        # Get true covariance matrix and set true weights
        if random=True:
                    corr = randomBlockCorr(nCols, nBlocks,
                     ↪  minBlockSize=minBlockSize)
        else:
             corr = formBlockMatrix(nBlocks, covSize, bCorr)

       cov0 = formTrueMatrix(corr)
       w0.loc[i] = md.allocate(cov0)

        # Get sample covariance matrix and set weights
        cov1 = simCov(cov0, nObs, shrink=shrink)
        #q = nObs/cov0.shape[0]
        #cov1 = pd.DataFrame(deNoiseCov(cov1, q, bWidth=.01))
        w1.loc[i] = md.allocate(cov1)
```

```
            # Single linkage
            w1_single.loc[i] = nco.allocate(cov1, linkage='single', max_k=15,
             ↪  w_method='md')
            k_single.append(nco.k)

            # Average linkage
            w1_average.loc[i] = nco.allocate(cov1, linkage='average', max_k=15,
             ↪  w_method='md')
            k_average.append(nco.k)

            # Ward's Method
            w1_ward.loc[i] = nco.allocate(cov1, linkage='ward', max_k=15,
             ↪  w_method='md')
            k_ward.append(nco.k)

            # Save estimated number of clusters (k)
            true_k.append(nBlocks)

        row = []

        # Calculate Root-Mean-Square Errors between weights
        row.append(np.mean((w1 - w0).values.flatten() ** 2 ) ** 0.5)
        row.append(np.mean((w1_single - w0).values.flatten() ** 2) ** 0.5)
        row.append(np.mean((w1_average - w0).values.flatten() ** 2) ** 0.5)
        row.append(np.mean((w1_ward - w0).values.flatten() ** 2) ** 0.5)

        row = pd.Series(row)
        stats = stats.append(row, ignore_index=True)

    stats.columns = ['MV', 'Single', 'Average', 'Ward']
    stats.index = ['Raw','Shrunk']
```

A4.2 Monte Carlo Backtest: Risk-Based Performance

```
# Import pre-generated correlation matrices
corr_matrix = pd.read_hdf('dataset/corrgan_corr.h5', 'corr')

# Set size of matrix
n = 100
a, b = np.triu_indices(n, k=1)

# list to store correlation matrices
corr_matrix_list = []

# Iterate through rows of flattened list
for row in corr_matrix.iterrows():
    flat_corr = row[1] # get row values
    flat_corr = flat_corr[::-1] # revert order of values
    corr = np.ones((n, n))  # fill a symmetric one matrix with size n
    corr[a, b] = flat_corr # input coefficients to top triangle
    corr[b, a] = flat_corr # input coefficients to bottom triangle
    corr_matrix_list.append(corr) # append to list

# Define Gaussian returns
def generate_returns_sample(cov, horizon=252):
    gaussian_ret = np.random.multivariate_normal(np.zeros(len(cov)), cov,
     ↪  size=horizon)
    return pd.DataFrame(gaussian_ret)

# Shuffle Matrix
def shuffle_matrix(matrix):
    matrix = pd.DataFrame(matrix)
    cols = matrix.columns.tolist()
    np.random.shuffle(cols)
    matrix = matrix[cols].loc[cols].copy(deep=True)
    return matrix
```

```
# Monte Carlo simulations
def monte_carlo(methods, iterations=10, in_horizon=252, out_horizon=252):

    # Initiate summary statistics dataframe
    vol = pd.DataFrame()
    dr = pd.DataFrame()
    ssw = pd.DataFrame()

    # Iterate through the methods
    for name, method in methods.items():

        # initiate return dataframes
        in_sample = pd.DataFrame()
        out_sample = pd.DataFrame()

        # initiate weighted volatility dataframes
        in_w_vol = pd.DataFrame()
        out_w_vol = pd.DataFrame()

        squared_weights = pd.DataFrame()

        for i in tqdm(range(iterations)):

            # generate in-sample and out-of-sample returns
            corr = pd.DataFrame(corr_matrix_list[i])
            cov = formTrueMatrix(corr)
            in_ret = generate_returns_sample(cov, horizon=in_horizon)
            out_ret = generate_returns_sample(cov, horizon=out_horizon)

            # Generate in-sample covariance for weight alloaction
            cov1 = in_ret.cov()

            # allocate weights
            if method[1] != None:
```

```
            args = method[1]
            try:
                weights = method[0].allocate(cov1, *args)
            except:
                pass
        else:
            try:
                weights = method[0].allocate(cov1)
            except:
                pass

        squared_weights[i] = (weights ** 2)

        # generate weighted volatilities for diversification ratio
        in_w_vol[i] = in_ret.std() * weights
        out_w_vol[i] = out_ret.std() * weights

        # generate portfolio returns and append to DataFrame
        in_sample[i] = (in_ret * weights).sum(axis=1)
        out_sample[i] = (out_ret * weights).sum(axis=1)

    # Get in-sample statistics
    in_vol = in_sample.std() * np.sqrt(252)
    in_dr = (in_w_vol * np.sqrt(252)).sum() / in_vol

    # Get out-of-sample statistics
    out_vol = out_sample.std() * np.sqrt(252)
    out_dr = (out_w_vol * np.sqrt(252)).sum() / out_vol

    squared_weights = squared_weights.sum()

    # Save volatilities
    vol[name + ' in'] = in_vol
    vol[name + ' out'] = out_vol
```

```
        # Save diversification ratios
        dr[name + ' in'] = in_dr
        dr[name + ' out'] = out_dr

        ssw[name + ' in'] = squared_weights
        ssw[name + ' out'] = squared_weights

    return vol, dr, ssw

# ------------------------------
# Testing
# ------------------------------

methods = {
    'MV': [mv, None],
    'MD': [md, None],
    'ERC': [erc, None],
    'IVP': [ivp, None],
    'NCO MV': [nco, [linkage, 50, 'mv']],
    'NCO MD': [nco, [linkage, 50, 'md']],
    'NCO ERC': [nco, [linkage, 50, 'erc']],
    'HRP': [hrp, [linkage, True, 'var']],
    'HERC': [herc, [linkage, 50, 'var']],
    'NCO MV': [nco, [linkage, 50, 'mv']],
    'NCO MD': [nco, [linkage, 50, 'md']],
    'NCO ERC': [nco, [linkage, 50, 'erc']],
    'HRP': [hrp, [linkage, True, 'var']],
    'HERC': [herc, [linkage, 50, 'var']],
    'NCO MV': [nco, [linkage, 50, 'mv']],
    'NCO MD': [nco, [linkage, 50, 'md']],
    'NCO ERC': [nco, [linkage, 50, 'erc']],
    'HRP': [hrp, [linkage, True, 'var']],
    'HERC': [herc, [linkage, 50, 'var']]
}

# Initiate Summary Stats DataFrame
summary_stats = pd.DataFrame()

# Run Monte Carlo
vol, dr, ssw = monte_carlo(methods, 2000, in_horizon=252*2, out_horizon=252)

# Save Summary Statistics
summary_stats['Vol'] = vol.mean()
summary_stats['DR'] = dr.mean()
summary_stats['SSW'] = ssw.mean()

# Output Summary Statistics
summary_stats
```

A4.3 Historical Backtest

```
def rebalance_weights(asset_returns, method, args=None, rebalance=3*21,
↪ window=2*252, testing='out-of-sample', shrink=True):
    # Initiate weights
    count = asset_returns.reset_index(drop=True) # get integer indice numbers
    weights = pd.DataFrame(index=count.index, columns=asset_returns.columns) #
    ↪ use integer indice for window calc

    # get rows used for rebalancing
    reb_index = weights.iloc[window::rebalance]

    # loop through rebalancing dates
    for i in tqdm(reb_index.index):
        # Restrict to timeframe and cleanse nan columns
        sample = asset_returns.iloc[i-window:i]
        sample = sample[sample.columns[~sample.isna().any()]]

        if shrink == True:
            cov = risk_matrix(sample, method='ledoit_wolf_constant_correlation',
             ↪  returns_data=True)
        else:
            cov = sample.cov()

        # allocate with length equal to window
        if args == None:
            weights.iloc[i] = method.allocate(cov)
        else:
            weights.iloc[i] = method.allocate(cov, *args)

    # forwardfill/backfill to get over rebalancing period
    if testing == 'out-of-sample':
        weights = weights.ffill(limit=rebalance-1)
    else:
        weights = weights.bfill(limit=rebalance-1)

    # set nan values to 0 for weight calculation
    weights = weights.fillna(0)
    weights.index = asset_returns.index

    return weights

# get dataframe from walkforward
def walk_forward(asset_returns, weights, warmup):
    ret = (asset_returns*weights)
    ret = ret.iloc[warmup:]
    return ret

# Cumulative returns
def cum_ret(returns):
    return (returns + 1).cumprod()[-1]

# Average returns
```

```python
def avg_ret(returns, freq='yearly'):
    if freq == 'yearly': freq = 252
    elif freq == 'monthly': freq = 21
    else: freq = 1
    return (1 + returns.mean())**(freq/1) - 1

# Volatility of returns
def vol_ret(returns, freq='yearly'):
    if freq == 'yearly': freq = 252
    elif freq == 'monthly': freq = 21
    return returns.std() * np.sqrt(freq)

def dr(asset_returns, weights, window=3*21):
    w_vol = (weights *
     ↪  asset_returns.shift(3*21).rolling(window).std()).sum(axis=1).mean()
    p_vol = (weights * asset_returns).sum(axis=1).std()
    return (w_vol / p_vol)

# Sharpe ratio
def sharpe(returns, rf=0):
    return ((returns - rf).mean() / (returns - rf).std()) * np.sqrt(252)

# Lower partial moment of the returns (for Sortino)
def lpm(returns, threshold, order=2):
    threshold_array = np.empty(len(returns))
    threshold_array.fill(threshold)
    diff = threshold_array - returns
    diff = diff.clip(0, None)
    return (np.sum(diff ** order) / len(returns))

# Sortino ratio
def sortino(returns, rf=0):
    return ((returns - rf).mean() / np.sqrt(lpm(returns, 0))) * np.sqrt(252)
```

```
# Value-at-risk
def var(returns, alpha):
    sorted_returns = np.sort(returns)
    index = int(alpha * len(sorted_returns))
    return abs(sorted_returns[index]) * np.sqrt(252)

# Expected Shortfall (cVaR)
def cvar(returns, alpha):
    sorted_returns = np.sort(returns)
    index = int(alpha * len(sorted_returns))
    sum_var = sorted_returns[0] # Calculate the total VaR beyond alpha
    for i in range(1, index):
        sum_var += sorted_returns[i]
    return abs(sum_var / index) * np.sqrt(252) # Return the average VaR

# Maximum Drawdown
def max_dd(returns):
    r = returns.add(1).cumprod()
    dd = r.div(r.cummax()).sub(1)
    mdd = dd.min()
    end = dd.idxmin()
    start = r.loc[:end].idxmax()
    return mdd, start, end

def portfolio_turnover(weights, warmup=252*2):
    weights = weights[252*2:]
    weights = weights.drop_duplicates()
    return (weights - weights.shift(1)).abs().sum(axis=1).mean()

def sspw(weights, rebalancing=3*21):
    weights = weights[252*2:]
    weights = weights.drop_duplicates()
    return (weights**2).sum(axis=1).mean()
```

```python
def rolling_sspw(weights, rebalancing=3*21):
    weights = weights[252*2:]
    weights = weights.drop_duplicates()
    return (weights**2).sum(axis=1)

def rolling_pto(weights, warmup=252*2):
    weights = weights[252*2:]
    weights = weights.drop_duplicates()
    return (weights - weights.shift(1)).abs().sum(axis=1).rolling(4).mean()
```

A5 Full Results Historical Backtest

	Average Yearly Return	**Average Yearly Volatility**
MV	13.4%	10.2%
NCO Single MV	13.4%	10.2%
MV Average NCO	12.5%	10.1%
MV Ward NCO	13.7%	10.2%
MD	18.0%	13.5%
MD Single NCO	19.4%	16.5%
NCO Average MD	14.8%	15.8%
MD Ward NCO	16.1%	15.2%
ERC	15.6%	15.9%
ERC Single NCO	18.2%	15.6%
ERC Average NCO	15.4%	15.3%
ERC Ward NCO	15.4%	15.9%
Single HRP	15.2%	14.4%
Average HRP	15.2%	14.4%
Ward HRP	15.3%	14.3
Single HERC	15.1%	14.6
Average HERC	14.8%	13.8%
Ward HERC	14.8%	14.3%
IVP	14.8%	15.1%
EW	16.3%	18.1%

Table A5.1: Return and Volatility for all Portfolios

	Skewness	**Kurtosis**	**Max**	**Min**	**5% quantile**	**95% quantile**
MV	0.09	15.39	8.7%	-6.3%	-0.9%	1.0%
NCO Single MV	0.05	16.94	9.4%	-6.2%	-0.9%	1.0%
MV Average NCO	-0.05	15.79	8.9%	-6.5%	-0.9%	1.0%
MV Ward NCO	-0.14	13.20	8.7%	-6.5%	-0.9%	1.0%
MD	0.22	16.11	12.8%	-7.9%	-1.2%	1.3%
NCO Single MD	37.80	18.32	16.4%	-10.4%	-1.5%	1.5%
NCO Average MD	-0.1639	7.9300	0.1247	-8.7%	-1.4%	1.5%
MD Ward NCO	-0.21	10.51	11.4%	-8.7%	-1.4%	1.4%
ERC	-0.21	11.15	10.90%	-9.1%	-1.5%	1.4%
ERC Single NCO	-35.4	8.03	0.1070	-9.0%	-1.4%	1.4%
ERC Average NCO	-32.1	11.10	0.1219	-9.4%	-1.4%	1.4%
ERC Ward NCO	-0.25	13.07	0.1246	-9.9%	-1.5%	1.4%
Single HRP	-0.20	11.35	0.1035	-8.3%	-1.4%	1.3%
Average HRP	-0.20	11.14	0.1036	-8.1%	-1.4%	1.3%
Ward HRP	-0.20	11.03	0.1030	-8.1%	-1.4%	1.3%
Single HERC	-0.07	15.87	0.1254	-8.8%	-1.3%	1.3%
Average HERC	-0.25	13.15	0.1065	-8.5%	-1.3%	1.3%
Ward HERC	-0.24	12.88	0.1102	-8.6%	-1.3%	1.3%
IVP	-0.16	11.51	0.1069	-8.7%	-1.4%	1.4%
EW	-0.19	9.53	0.1113	-9.9%	-1.7%	1.7%

Table A5.2: Return Distribution of all Portfolios

	Maximum Drawdown	**VaR (5%)**	**CVaR (5%)**	**Sharpe**	**Sortino**
MV	-30.2%	14.7%	23.2%	0.94	1.39
NCO Single MV	-32.1%	14.8%	23.6%	0.93	1.37
MV Average NCO	-33.5%	14.4%	23.6%	0.86	1.25
MV Ward NCO	-31.3%	14.7%	23.7%	0.95	1.40
MD	-51.2%	19.3%	30.6%	0.99	1.47
MD Single NCO	-61.0%	22.2%	35.7%	0.88	1.32
NCO Average MD	-63.1%	22.0%	36.2%	0.67	0.98
MD Ward NCO	-50.6%	22.5%	35.9%	0.77	1.11
ERC	-52.5%	23.3%	37.9%	0.72	1.0444
MD Single NCO	-61.0%	23.8%	36.6%	0.89	1.32
ERC Average NCO	-55.9%	22.2%	35.7%	0.73	1.05
ERC Ward NCO	-53.9%	23.0%	37.7%	0.71	1.03
Single HRP	-49.3%	21.5%	34.4%	0.78	1.12
Average HRP	-49.2%	21.6%	34.4%	0.77	1.12
Ward HRP	-49.0%	21.5%	34.3%	0.78	1.14
Single HERC	-54.1%	20.9%	33.3%	0.74	1.08
Average HERC	-51.5%	20.2%	32.5%	0.77	1.12
Ward HERC	-49.6%	20.8%	34.1%	0.75	1.09
IVP	-49.9%	22.3%	36.1%	0.72	1.05
EW	-55.7%	26.6%	43.1%	0.67	0.97

Table A5.3: Risk-Based Performance-Metrics for all Portfolios

	Portfolio Turnover	**SSPW**	**Avg. Max Weight**
MV	46.5%	0.0593	12.3%
NCO Single MV	47.4%	0.0551	11.7%
MV Average NCO	60.4%	0.0942	17.3%
MV Ward NCO	61.4%	0.0477	10.5%
MD	49.2%	0.0357	8.4%
MD Single NCO	93.1%	0.0835	15.1%
NCO Average MD	108.7%	0.0521	12.6%
MD Ward NCO	111.8%	0.0228	5.9%
ERC	6.6%	0.0021	0.7%
ERC Single NCO	93.2%	0.0656	14.1%
ERC Average NCO	67.2%	0.0186	7.3%
ERC Ward NCO	45.9%	0.0034	1.2%
Single HRP	26.7%	0.0034	1.7%
Average HRP	27.6%	0.0031	1.1%
Ward HRP	27.5%	0.0031	1.1%
Single HERC	74.7%	0.0638	15.4%
Average HERC	68.2%	0.0317	9.1%
Ward HERC	47.8%	0.0050	2.1%
IVP	7.7%	0.0027	0.9%
EW	2.6%	0.0017	0.2%

Table A5.4: Weight Concentration and Asset Turnover for all Portfolios