University of Birmingham

Realised Semibetas

Mayank Juyal

2692774

Msc. Fiancial Engineering

School of Mathematics

Sept 2024

Abstract

This dissertation investigates the predictive power of semibetas in explaining asset returns, comparing them to traditional asset pricing models such as the Capital Asset Pricing Model (CAPM) and the Fama-French factors. The study decomposes the market beta into four distinct semibetas, and tests them in asset pricing models. The results reveal that semibetas, particularly those related to negative market movements, offer a better explanation of asset returns than conventional models. It suggest that incorporating semibetas into asset pricing models can significantly enhance their predictive power, offering insights for investors and researchers in portfolio management.

Contents

1 Introduction

Asset pricing models are used to understand how different factors impact the returns of various financial assets. These models enable the investors to make a more informed decision, by helping them in quantifying the relationship between return and risk. The ability of these models to incorporate various risk factors that impact asset prices, can help investors to optimize their portfolios. For example, one of the traditional asset pricing model , the Capital Asset Pricing Model (CAPM)Sharpe (1964) uses an asset's sensitivity to the market returns to determine its expected return. Other models like Fama and French 3 Factor Model Fama & French (1993) use multiple factors like size and book value to achieve the same. These models help in estimating expected returns of the assets by accounting for various risk premiums, that an investor should expect for taking those risks, providing a more structured view on risk and reward.

The Capital Asset Pricing Model (CAPM), Sharpe (1964), establishes a simple linear relationship between an asset's return and its market risk, which is measured by beta (β). Although widely adopted, traditional CAPM has faced some empirical challenges, particularly in its ability to explain the cross-sectional variation in asset returns. This has sparked discussion on whether the traditional market beta is adequate for pricing financial assets. While some researchers like Fama et al. (1969) provided support for CAPM, subsequent literature like Roll (1977), Bhandari (1988), and Fama & French (1992) questions CAPM's ability to capture market risk.

CAPM's reliance on assumptions, such as normally distributed asset returns and a constant correlation between an asset's return and the market return limits its ability to capture real world conditions. They are crucial for simplifying the model and allowing the single-factor market beta β to capture market risk but do not reflect real world scenario. The assumption of normality suggests that extreme positive or negative events in asset returns are equally probable. However, research by Mandelbrot (1963) demonstrated that asset returns do not follow a normal distribution; instead, they exhibit heavy tails, indicating that extreme events occur more frequently than predicted by a normal distribution. This insight challenges the validity of CAPM, as it suggests that the model may underestimate the true risk associated with asset returns. Moreover, the assumption of a constant correlation between asset and market returns is often violated, especially during periods of financial stress. The study by Longin & Solnik (2001) has shown that correlations tend to increase during market downturns, indicating that assets become more closely linked during crises than in normal conditions. This implies that in times of financial turbulence, portfolios constructed under the assumption of constant correlation could face higher risks than anticipated. These findings cast doubt on the adequacy of CAPM's single-factor framework, and indicate scope of improvement.

Various researchers have attempted to address the shortcomings of the traditional CAPM. Studies such as those by Hogan and Warren (1974) have emphasized that investors tend to focus more on the potential for losses, or downside risk, compared to potential gains. They proposed that separating market variance into semivariances, distinguishing between positive and negative market returns, offers a more accurate approach in asset pricing models, especially since downside risk better aligns with investor concerns. Their research challenges the assumption that investors assess risk based solely on the total variance of an asset's returns in relation to the market, arguing instead that factors like downside risk and return asymmetry should be considered, elements that the traditional mean-variance framework of the Capital Asset Pricing Model (CAPM) fails to incorporate. This notion of investors' emphasis on downside risk is consistent with the concept of loss aversion proposed by Kahneman & Tversky (1979), which posits that investors tend to exhibit risk-averse behavior when facing the possibility of significant losses but are inclined toward risk-taking when dealing with low-probability, substantial gains.

The limitations of traditional asset pricing models have led to the exploration of alternative measures that might better capture the non-linear dependencies in asset returns. One such advancement is the concept of semibetas, introduced by Bollerslev et al. (2022), which offers a more granular decomposition of market beta. In their model, instead of a single market beta, as considered in the Capital Asset Pricing Model, they considered four semibetas. They showed that the traditional market beta could be decomposed into four components, based on the signs of the market returns and the returns of individual assets. Two of these semibetas correspond to positive market returns, while the other two correspond to negative market returns. They observed that the semibetas associated with positive market returns had low significance, whereas those associated with negative market returns showed significance in explaining asset returns. These negative market semibetas are further broken down into two more semibetas based on signs of asset returns. One measures covariation of negative asset returns with negative market returns and the other measures covariation of positive asset returns with negative market returns. It was found that the latter indicated lower future returns. Conversely, the former indicated higher future returns. Their findings suggest that investors may achieve better outcomes by strategically focusing on these semibetas, rather than relying solely on the traditional beta approach.

The objective of this dissertation is to critically assess the traditional CAPM framework, particularly focusing on its limitations in accounting for downside risk. It aims to delve into the concept of semibetas, examine their effectiveness in explaining cross-sectional variations in asset returns, and compare their predictive power against other established risk factors. Given that the CAPM largely depends on total variance as the sole measure of systematic risk, it often fails to address the asymmetry in asset returns. This study, therefore, seeks to move beyond the traditional beta, exploring additional factors influencing asset returns.

The dissertation is structured as follows: Section 2 starts with an introduction to traditional asset pricing models like Modern Portfolio Theory (MPT), introduced by Markowitz (1952), and CAPM developed by Sharpe (1964). It then discusses the underlying assumptions of these models, such as the normal distribution of asset returns, perfect markets, and highlights the limitations of these models, including the assumption of constant correlation between assets and market. This will set the stage for exploring models that aim to address these limitations

In Section 3, we will introduce semibetas by first explaining the idea of realized semicovariancesBollerslev et al. (2020). This concept helps break down an asset's covariance with the market into different parts based on positive and negative returns. These semicovariances are then used to calculate semibetas.

In Section 4, we will outline the methodology used in this study. We will describe the process of cleaning the data, constructing the market index, and calculating semibetas. This section will also introduce other models that are later used to compare against the semibetas to understand their effectiveness.

In Section 5, we will present our results, showing that semibetas are significant in explaining asset returns, of which semibeta related to negative market and asset movements is the most significant. This study finds that inclusion of semibetas in asset pricing models can increase their predictive power significantly.

2 Traditional Asset Pricing Models and Limitations

In this section, we will examine two fundamental models of market risk: the Modern Portfolio Theory (MPT) introduced by Markowitz (1952) and the Capital Asset Pricing Model (CAPM) developed by Sharpe (1964). Since the 1950s, the increasing availability of data on traded securities has led to significant growth in academic research on market risk, resulting in the empirical evaluation of numerous market risk models across various global markets. Both MPT and CAPM are grounded in assumptions that simplify complex market behaviors, enabling them to provide insights into key factors and their interrelationships. Although these assumptions help create a clear framework for understanding the markets, they often overlook the complexities of real world financial markets. As we examine these models further, we will discuss how empirical evidence exposes the limitations of MPT and CAPM.

2.1 Modern Portfolio Theory (MPT)

In his Ph.D. dissertation, which later earned him the Nobel Prize, Markowitz (1952) introduced what is now known as Modern Portfolio Theory (MPT). MPT provides a framework for investors to construct an optimal portfolio by selecting assets based on the expected returns of the portfolio, risk (measured by the portfolio's volatility), and the correlation of returns among different assets.

The theory demonstrates that through diversification investors can effectively eliminate idiosyncratic or company-specific risk, leaving only systematic risk[1], which represents the inherent risk of the overall market that cannot be diversified away. This focus on systematic risk allowed Markowitz to simplify the investor's utility function to depend solely on the mean (expected return) and variance (risk) of the portfolio's return, establishing mean-variance analysis as a central concept in portfolio optimization.

MPT operates on the assumption that investors behave rationally and aim to balance potential returns against risks. According to Markowitz, a 'rational investor,' who is risk-averse and seeks to maximize utility, should structure their portfolio by considering both the mean (expected return) and the variance (risk) of the portfolio. Faced with investment choices that offer the same expected return, such an investor would logically prefer the option with the least risk or variance. This concept leads to 'efficient frontier' which will be discussed in the following section.

2.1.1 Assumptions of MPT

Modern Portfolio Theory (MPT) operates under the assumption that capital markets are ideal, meaning there are no taxes or transaction costs, information is freely and simultaneously available to all market participants, and that all investors act rationally in a perfectly competitive market environment. Additionally, it assumes that asset returns follow a normal distribution. The assumption of normally distributed returns enables investors' utility choices to be expressed in terms of mean (expected return) and variance (risk).

According to MPT, the proportion invested in each asset within a portfolio should be determined by its contribution to the overall portfolio's return distribution, specifically its impact on both return and risk of the overall portfolio. In this context, the performance of an individual asset is less significant than how it interacts with other assets in the

[1] Systematic risk refers to market-wide risks that affect all assets, while idiosyncratic risk—also known as unsystematic or diversifiable risk—is specific to an individual asset. For instance, in the context of a stock, idiosyncratic risk might stem from company-specific events like labor strikes, regulatory changes, or reputation issues and systematic risk stems from the risk of investing in the financial markets as a whole.

portfolio. Essentially, the covariability with portfolio is more critical than the asset's standalone return.

Generally, there is a positive relationship between risk and expected return, with higher risk leading to higher expected returns. However, by diversifying and appropriately allocating assets, investors can achieve an optimal return for a given level of risk. This concept gives rise to the "efficient frontier," which represents a curve depicting the set of portfolios that offer the highest expected return for each level of risk.

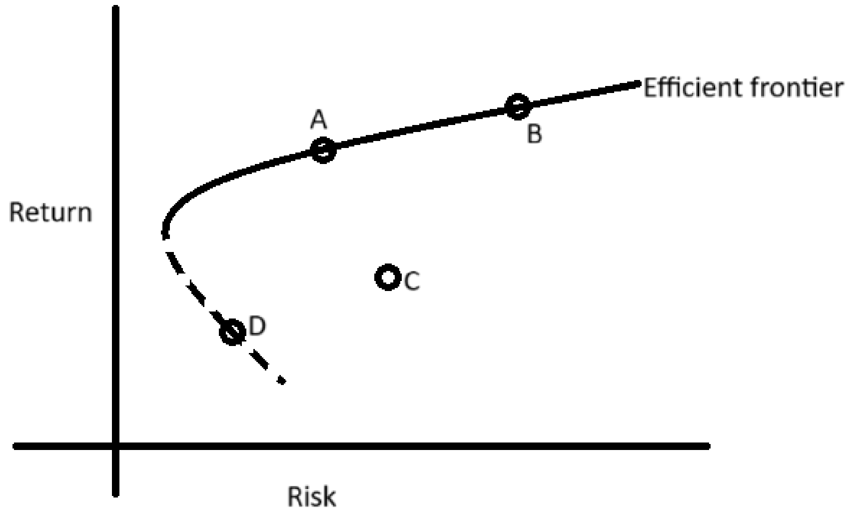

Figure 1: The efficient frontier showing the relationship between risk and expected return. Portfolios A and B represent optimal choices on the frontier, while Portfolio C is suboptimal for a given level of risk. The dotted line indicates inefficient portfolios (D) with lower returns relative to the risk undertaken.

In Figure 1, is a hypothetical representation for efficient frontier. The solid curve represents the efficient frontier, portfolio A, B signifies the portfolio that offers the highest possible expected return at a given level of risk, positioning it as an optimal choice on the efficient frontier. On the other hand, Portfolio C represents a less optimal choice because another portfolio exists that can yield a higher return for the same level of risk. The dotted line indicates portfolios (D) that are inefficient, where the investor receives lower return relative to the level of risk undertaken. As one progresses along the efficient frontier—which is essentially the curve representing the portfolios with the best possible expected returns for each risk level—increasing the expected return necessitates a corresponding increase in risk.

Another significant contribution from Markowitz was in of asset correlations. The correlation between assets allow an investor to construct a portfolio that balances adverse affects on a particular asset with favorable affects on other assets. To achieve this a constant correlation is assumed between asset and market returns.

While Modern Portfolio Theory (MPT) played a significant role in portfolio selection, it has faced criticism regarding some of its assumptions and practical applications. Empirical studies have shown that asset returns do not follow a normal distribution; instead, they often exhibit "fat tails" as highlighted by studies from Mandelbrot (1972) and Fama (1965). Another key assumption of MPT is its reliance on mean and variance, without accounting for skewness in the return distribution. The theory also does not provide guidance on which historical data period to use when estimating the model's parameters, such as the mean, variance of returns, and correlations between assets. Choosing different periods can result in significantly different parameter values as shown byMichaud (1989).

2.2 CAPM: The Capital Asset Pricing Model

The Capital Asset Pricing Model (CAPM) was first introduced in a paper by Sharpe (1964), and later expanded upon by Lintner (1965) and Mossin (1966). It laid the groundwork for asset pricing theory and eventually led to Sharpe being awarded the Nobel Prize in 1990. Even decades after its introduction, CAPM continues to be a fundamental tool in portfolio management and asset pricing.

The Capital Asset Pricing Model (CAPM) advanced the concepts of Modern Portfolio Theory (MPT) by introducing the concept of capital market equilibrium. In the context of CAPM, capital market equilibrium refers to a state where all securities in the market are fairly priced, meaning that their current market prices accurately reflect all available information about their risk and return. In this equilibrium state, the expected returns of all assets are aligned with their level of systematic risk, as measured by their sensitivity to market movements (measured by β).

In such a market equilibrium, investors cannot achieve excess returns without taking on additional risk, as all opportunities for arbitrage have been eliminated. This means that the prices of assets adjust to a point where supply equals demand, and all investors, regardless of their risk preferences, hold a combination of the market portfolio and a risk-free asset. Therefore, the CAPM suggests that in an efficient and balanced market, the only way to increase expected returns is to accept a higher level of systematic risk.

Sharpe (1964) and Lintner (1965) introduced two key assumptions to the Markowitz's Modern Portfolio Theory (MPT). The first assumption is the notion of unlimited borrowing and lending at a risk-free rate, allowing all investors to borrow or lend funds at this rate regardless of the amount. The second assumption is that of homogeneous expectations, meaning all investors share identical expectations regarding future asset returns. They concur on the joint probability distribution of asset returns, which serves as the basis for testing the model. This assumption suggests that all investors possess and interpret information in a uniform manner, with risk and return being evaluated consistently, and prices reflecting all available information. Although this assumption streamlines the model, it has faced criticism for being unrealistic, as real-world investors often have diverse information and expectations, we will discuss these criticisms along with other limitations in following section.

The CAPM Equation

The CAPM equation is commonly represented as:

$$\text{Expected Return}_i = \text{Risk-free rate} + \beta_i \times \text{Market Risk Premium}$$

Here,i represents an asset, β_i represents the asset's sensitivity to market movements, and the Market Risk Premium represents the excess return of the market over the risk-free rate. The beta of the asset β_i is calculated as:

$$\frac{\text{Cov}(\text{Asset}_i, \text{Market})}{\text{Variance of Market}}$$

The total risk of an asset, represented as σ_i, can be divided into systematic risk, expressed as $\beta_i \sigma_M$, and specific or idiosyncratic risk.

$$\sigma_i^2 = \beta_i^2 \sigma_M^2 + \sigma_{\text{idiosyncratic}}^2 \tag{1}$$

In a well-diversified portfolio, the idiosyncratic risk is minimized or effectively eliminated, meaning that only systematic risk influences the portfolio's return. Consequently, investors are rewarded for bearing systematic risk only.

The expected return on an asset is determined by adding the risk-free rate to a risk premium, which is calculated as the product of the asset's beta and the market risk premium. Therefore, assets with a higher beta are expected to yield higher returns.

2.2.1 Limitations of CAPM

Despite the widespread adoption of CAPM, researchers like Roll (1977) and others have found several limitations in CAPM, which shows that the assumptions of CAPM constraints its applicability in real-world scenarios. We will discuss some of these limitations in this section

Assumptions and Real-World Constraints

Homogeneous Expectations: One of the assumptions of CAPM is homogeneous expectations, which implies that all investors have identical expectations regarding returns, variances, and covariances of assets. Merton et al. (1987)[2] showed that differences in information can lead to different equilibrium outcomes in the market. He showed that incomplete information can lead to equilibrium prices that reflect the differential information possessed by investors. This shows that in reality, information in the market is

[2]Merton explored the impact investor's expectations on capital market equilibrium, showing how homogeneous expectations might not hold in reality.

diverse, and investors have heterogeneous expectations. This discrepancy can lead to differences in portfolio choices made by investors and asset pricing.

Market Efficiency: An assumption of CAPM is that markets are efficient, meaning all available information is instantly incorporated into asset prices. However, research by De Bondt & Thaler (1985) indicates that factors such as investor sentiment, information asymmetry, and behavioral biases can lead to market inefficiencies, suggesting that prices do not always fully reflect all relevant information.

Single Period Investment Horizon: The CAPM's assumption of a single investment horizon oversimplifies the reality of investment decisions, as many investors plan over multiple periods. Merton (1973) sought to address this limitation by introducing multi-period models, such as the Intertemporal Capital Asset Pricing Model (ICAPM), which take into account the planning and decision-making of investors over varying time horizons.

Empirical Inconsistencies

Failure to Explain Cross-Sectional Returns: The CAPM has limitations in explaining variation in cross sectional returns of assets, as it relies solely on beta (β) to measure systematic risk. Research by Fama & French (1992) challenged this approach by demonstrating that factors such as firm size and book-to-market ratios play a significant role in explaining asset returns. Fama and French extended the CAPM by incorporating additional size and value factors, thereby providing a more comprehensive explanation of asset returns. their model is commonly represented as: Expected return of asset = Risk-free rate + Beta of asset × (Expected market return - Risk-free rate) + Sensitivity to size factor × SMB + Sensitivity to value factor × HML, where SMB (Small Minus Big) represents the difference in returns of small vs large companies, capturing the risk associated with small-cap versus large-cap stocks, and HML (High Minus Low) indicates the return difference between high book-to-market (value) and low book-to-market (growth) stocks. This extended model shows that their are additional factors that need to be considered for explaining the cross-sectional variation in returns which have not been included in CAPM.

Low Predictive Power of Beta: The CAPM's assumption of a strong relationship between beta and expected returns has proven to be weaker than anticipated. In some situations, the risk premiums associated with beta are found to be too low. According to Roll (1977), one of the main challenges is that the true market portfolio is unobservable[3], which undermines the empirical testing of CAPM. Roll pointed out that the actual cor-

[3]because the true market portfolio would include all types of assets in the economy—such as stocks, bonds, real estate, private businesses, human capital, and other forms of wealth—that are not entirely measurable or accessible for analysis.

relation between beta and expected returns is often weaker than what CAPM suggests. Additionally, Bhandari(1988) found evidence that companies with higher debt-to-equity ratios tend to exhibit higher expected returns, indicating that leverage and other factors also play a crucial role in determining asset returns.

Downside Risk Considerations: Traditional CAPM does not distinguish between downside and upside risk. Research by Ang et al. (2006) suggests that investors are particularly averse to downside risk. The downside beta version of CAPM, based on negative return covariation, often explains cross-sectional returns better than traditional CAPM. They found that stocks with higher exposure to downside risk, or those with higher downside beta, tend to have higher expected returns.

Arbitrage and Liquidity Constraints: The assumption of no arbitrage was also found to be limiting. Pontiff (1996)'s paper examines the costs and limits associated with arbitrage, highlighting the constraints in real markets. The paper uses closed-ended funds to examine the costs and limits associated with arbitrage. It highlights that arbitrage is not always riskless or costless. Transaction costs, short-selling constraints, and other market frictions can create significant barriers to arbitrage. This can lead to different pricing of risk for long and short positions, affecting the risk premiums.

2.3 Need for Improvement in Asset Pricing Models

Normal Distribution: Most traditional asset pricing models assume returns to be normally distributed. This implies that in a univariate distribution, the probability of extreme events, positive or negative, is the same. In multivariate distributions, asset returns are assumed to be jointly normally distributed.Mandelbrot (1963) provided evidence challenging this claim and showed that asset returns instead have a heavy tail. Since then, market data has repeatedly confirmed this claim of the non-Gaussian character of asset returns. One way to describe non-normal distributions is kurtosis, defined by a parameter k, where $k = 0$ indicates a normal distribution, while a positive value of k indicates fat tails. Asset returns were found to be non-Gaussian, heavy-tailed, and sharp-peaked. This implies that large price changes or extreme events have a higher probability than predicted by a normal distribution. Fama (1965) also showed similar characteristics of return distribution, fat tails, and skewness.

Constant correlation: While an asset's return distribution can be represented by a univariate distribution, a portfolio return is represented by a multivariate distribution, which describes the behavior of multiple assets simultaneously, including how they behave together. Traditional asset pricing models like CAPM assume that the correlation between assets is a constant. This implies that the asset correlations are same during periods of market crashes and market upturns. Researchers like Patton (2004) and longin (2001) highlight that in practice, this assumption of symmetric dependencies is violated.

? longin (2001) showed that the correlation between international equity markets is not constant and tends to increase during market downturns. This phenomenon is known as correlation breakdown. They found that the correlations between different equity markets behave differently depending on the market conditions. They may remain stable during normal or bull market conditions but rise sharply during market crashes. This implies that a portfolio that may seem diversified in normal times may become highly correlated during market downturns, reducing the diversification benefits when it is needed most. This implies that the risk of large, simultaneous losses will be underestimated by models assuming constant correlations. The phenomenon where the correlation between asset returns increases when returns exceed a certain threshold limit is called exceedance correlation and their study revealed that when returns fall below a certain negative threshold, the correlation between equity markets intensifies more than when returns exceed a positive threshold.

The limitation of traditional models, which assume normally distributed returns and constant correlations across market conditions, indicates the scope of improvement. Research has indicated that there may be other factors required for explaining asset returns.

3 Semibeta Decomposition

Ang et al. (2006) discovered that the if the market beta is broken down into two, one representing asset's covariation with positive market movements and the other with negative market movements, then we can explains variation in U.S. equity returns better, compared to the traditional CAPM. These decomposed betas are called upside and downside betas referring to the market upturns and downturns respectively.

To account for this effect, Bollerslev et al. (2022) introduced a method to decompose the conventional market beta into four distinct semibetas. This approach builds on the concept of semicovariance as outlined by Bollerslev et al. (2020).

Bollerslev et al. (2020) research explores some interesting questions like, do variances and covariances react differently to positive vs. negative returns? Do investors process positive and negative returns differently? Is there information in signed returns?

Variance measures how an asset's returns deviate from its average return, while covariance measures how different assets move together. Bollerslev et al. (2020) explored whether variances and covariances change based on market returns being positive or negative, i.e., during market uptrends and downtrends. Kahneman & Tversky (1979) showed that market participants react differently to positive returns compared to negative returns. The market participants show more sensitivity to losses than to gains; this concept is known as loss aversion. Bollerslev et al. (2020) also found that there is significant information in the signs of returns.

3.1 Realized SemiCovariances

In this section, we will explore the work of Bollerslev et al. (2020) to understand the process of decomposing semicovariances of asset with market returns. The formulas and equations presented in this section are sourced from their paper, "Realized Semicovariances". The concept of realized semicovariance (semicovariance of intraday high frequency asset returns with market returns) extends from the traditional notion of realized covariance (covariance of intraday high-frequency asset returns with market returns). This decomposition divides the realized covariance into three separate realized semicovariance components based on the signs of returns. The formula for calculating realized covariance is as follows:

$$\mathrm{RCOV}_t^{(m)} = \sum_{l=1}^{m} r_{l,t} r'_{l,t} \tag{2}$$

Bollerslev et al. (2020)

where $r_{l,t}$ and and $r'_{l,t}$represents a vectors of returns for different stocks over the l-th interval of the day t.

$$\mathbf{r}_{l,t} = [r_{1,l,t}, \ldots, r_{N,l,t}]' \tag{3}$$

The matrix $r_{l,t} r'_{l,t}$ represents the outer product of the return vectors at interval l of day t. By summing these outer products over all m high-frequency intervals, we obtain the realized covariance for the entire day.

3.1.1 Vector of Positive and Negative Returns

To compute semicovariances, the return vector is divided into two parts: one containing the positive returns and the other containing the negative returns. These components are defined as follows:

$$r_{l,t}^{+} = r_{l,t} \quad \text{where} \quad \{r_{l,t} \geq 0\} \tag{4}$$

$$r_{l,t}^{-} = r_{l,t} \quad \text{where} \quad \{r_{l,t} < 0\} \tag{5}$$

Here, $r_{l,t}^{+}$ is a vector that contains only positive returns, all negative returns are set to zero, and $r_{l,t}^{-}$ is a vector that has only negative returns, all positive returns are set to zero.

3.1.2 Realized Semicovariance Matrices

Using the positive and negative return vectors, we can define four types of semicovariances:

1. Positive-Positive Semicovariance, $P_t^{(m)}$: The positive-positive semicovariance represents the covariance between the assets when both have positive returns. It is represented as the sum of outer products of positive return vectors over m frequency intervals.

$$P_t^{(m)} = \sum_{l=1}^{m} r_{l,t}^{+}(r_{l,t}^{+})' \quad \text{(Positive-positive semicovariance)} \tag{6}$$

Bollerslev et al. (2020)

2. Negative-Negative Semicovariance, $N_t^{(m)}$: The negative-negative semicovariance represents the covariance between the assets when both have negative returns. It is represented as the sum of outer products of negative return vectors over m frequency intervals.

$$N_t^{(m)} = \sum_{l=1}^{m} r_{l,t}^{-}(r_{l,t}^{-})' \quad \text{(Negative-negative semicovariance)} \tag{7}$$

Bollerslev et al. (2020)

3. Positive-Negative Semicovariance, $M_t^{+(m)}$ and the Negative-positive Semicovariance, $M_t^{-(m)}$: These represent the covariance between the assets when both have opposite returns. It is represented as below

$$M_t^{+(m)} = \sum_{l=1}^{m} r_{l,t}^{+}(r_{l,t}^{-})' \quad \text{(Positive-negative semicovariance)} \tag{8}$$

Bollerslev et al. (2020)

$$M_t^{-(m)} = \sum_{l=1}^{m} r_{l,t}^{-}(r_{l,t}^{+})' \quad \text{(Negative-positive semicovariance)} \tag{9}$$

Bollerslev et al. (2020)

The sum of $M_t^{+(m)}$ and $M_t^{-(m)}$ forms the symmetric matrix $M_t^{(m)}$, which represents the mixed semicovariances.

These semicovariance matrices capture behaviors between assets under different market conditions. The realized covariance matrix has now been decomposed into different components, and the total realized covariance is the sum of these components.

$$\mathrm{RCOV}_t^{(m)} = P_t^{(m)} + N_t^{(m)} + M_t^{(m)} \tag{10}$$

Bollerslev et al. (2020)

where $M_t^{(m)}$ represent mixed semicovariances, $P_t^{(m)}$ and $N_t^{(m)}$ represent positive and negative semicovariances.

3.1.3 Bivariate Case of Semicovariances

The bivariate cases of the semicovariance can be represented as:

$$P_t^{(m)} = \begin{pmatrix} \sum_{l=1}^{m}(r_{1,l,t}^{+})^2 & \sum_{l=1}^{m} r_{1,l,t}^{+} r_{2,l,t}^{+} \\ \sum_{l=1}^{m} r_{2,l,t}^{+} r_{1,l,t}^{+} & \sum_{l=1}^{m}(r_{2,l,t}^{+})^2 \end{pmatrix} \tag{11}$$

$$N_t^{(m)} = \begin{pmatrix} \sum_{l=1}^{m}(r_{1,l,t}^{-})^2 & \sum_{l=1}^{m} r_{1,l,t}^{-} r_{2,l,t}^{-} \\ \sum_{l=1}^{m} r_{2,l,t}^{-} r_{1,l,t}^{-} & \sum_{l=1}^{m}(r_{2,l,t}^{-})^2 \end{pmatrix} \tag{12}$$

Bollerslev et al. (2020)

The positive semicovariance matrix $P_t^{(m)}$ is a case of a semicovariance matrix that focuses solely on the positive returns of the assets. The elements, $\sum_{l=1}^{m}(r_{1,l,t}^{+})^2$ and $\sum_{l=1}^{m}(r_{2,l,t}^{+})^2$, represent the variance between positive returns of asset 1 and 2. The elements, $\sum_{l=1}^{m} r_{1,l,t}^{+} r_{2,l,t}^{+}$ and $\sum_{l=1}^{m} r_{2,l,t}^{+} r_{1,l,t}^{+}$, represent the semicovariances between the two assets when the returns of both assets are positive. $P_t^{(m)}$ can be represented as matrix of semivariances and semicovariances.

$$P_t^{(m)} \equiv \begin{pmatrix} V_{1,t}^{+} & P_t^{+} \\ P_t^{+} & V_{2,t}^{+} \end{pmatrix} \tag{13}$$

Similarly, the negative semicovariance matrix $N_t^{(m)}$ is a case of a semicovariance matrix that only considers negative returns of the assets. Diagonal elements represents semivariances of the assets (only considering the negative returns of assets). Off diagonal elements represent the semicovariances between the two assets when returns of both are negative, i.e., the negative semicovariances.

$$N_t^{(m)} \equiv \begin{pmatrix} V_{1,t}^{-} & N_t^{-} \\ N_t^{-} & V_{2,t}^{-} \end{pmatrix} \tag{14}$$

3.1.4 Mixed Semicovariance Matrix

The $M_t^{+(m)}$ mixed semicovariance matrices can be represented as:

$$M_t^{+(m)} = \begin{pmatrix} \sum_{l=1}^{m} r_{1;l;t}^{+} r_{1;l;t}^{-} & \sum_{l=1}^{m} r_{2;l;t}^{+} r_{1;l;t}^{-} \\ \sum_{l=1}^{m} r_{1;l;t}^{+} r_{2;l;t}^{-} & \sum_{l=1}^{m} r_{2;l;t}^{+} r_{2;l;t}^{-} \end{pmatrix} \tag{15}$$

Bollerslev et al. (2020)

The off-diagonal entries of the mixed semicovariance matrix $M_t^{+(m)}$ measure the covariation between asset returns when they have opposite signs. If the asset is arbitrary (meaning there is no inherent reason to treat one asset as the first and the other as the second), it makes sense to combine the mixed matrices into a single symmetric matrix.

$$M_t^{(m)} = M_t^{+(m)} + M_t^{-(m)} \tag{16}$$

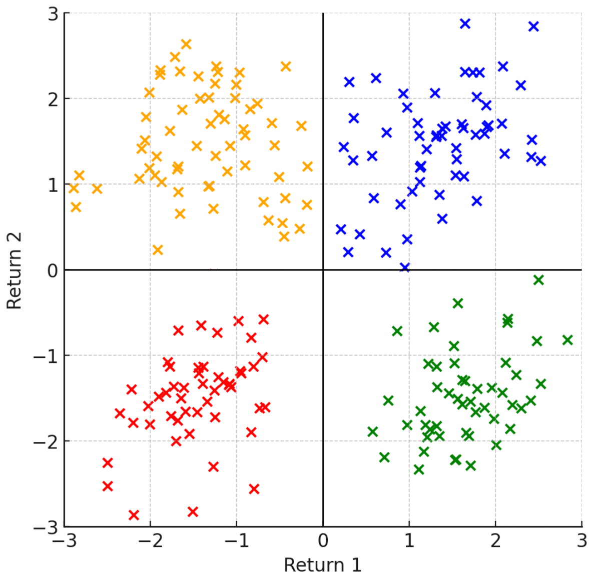

Figure 2: Hypothetical scatter plot of returns for two assets.

Figure 2 shows a hypothetical scatter plot of returns of two assets. The first (top right) and the third (bottom left) quadrants capture cases when the asset returns of both assets are positive and negative, respectively. The second (top left) and fourth (bottom right) quadrants capture mixed cases. Through decomposition, we measure covariation in each quadrant separately. The first and the third quadrants provide information on how assets co-move in bullish and bearish markets, respectively, while the second and fourth quadrants provide information on mixed interactions, which can help in identifying hedging opportunities.

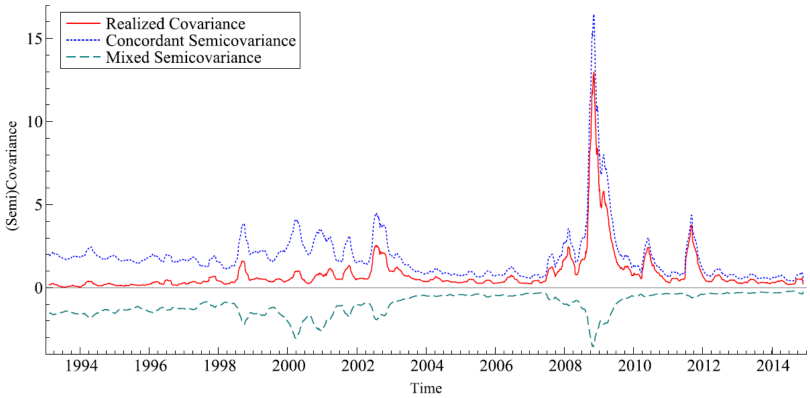

Figure 3: Time series of realized covariance decomposition. Source: Patton, A.J., Bollerslev, T., Li, J., Quaedvlieg, R. (2020). Realized Semicovariances. *Econometrica*, 88(4), 1515–1551.

In Figure 3 Bollerslev et al. (2020) shows results of their research, a time series plot of realized covariances and semicovariances across 500 pairs of stocks from January 1993 to December 2014. The P+N (Red) line represents the sum of the positive and negative semicovariances, RCOV (blue) line shows the traditional covariance, and M (green) line represents mixed semicovariances. They showed that the P+N (red) line stays above the RCOV line, suggesting that the correlation between assets is stronger when both assets are moving in the same direction, either up or down. The P & N line gradually increases, with noticeable spikes during periods of market stress. These spikes indicate increased co-movement of stocks during times of market stress. This is an important observation for risk management, as it highlights that during bull or bear markets, diversification benefits might be reduced due to increased correlation. The blue line is more stable but shows significant peaks during times of financial stress, particularly around 2008. This increase in the overall correlation during crises suggests that stocks tend to move more in unison during turbulent periods, indicating increased systemic risk. The green line, representing mixed semicovariances, also shows spikes during market stress, indicating potential increases in hedging during stress.

3.2 Semibeta Decomposition

Bollerslev et al. (2022) proposed decomposing the market beta, based on the concept of semicovariances discussed in the previous section. This decomposition is discussed in this section. Formulas presented in this section, are sourced from their paper to ensure consistency and accuracy.

The traditional beta is defined as:

$$\beta = \frac{\mathrm{Cov}(r, m)}{\mathrm{Var}(m)} \tag{17}$$

where r and m represent an asset's return and market returns. Bollerslev et al. (2022) decomposed β into four distinct components: β_N, β_P, β_{M^+}, and β_{M^-}
where,

$$\beta_N = \frac{N}{\mathrm{Var}(m)}, \quad \beta_P = \frac{P}{\mathrm{Var}(m)}, \quad \beta_{M^+} = \frac{M^+}{\mathrm{Var}(m)}, \quad \text{and} \quad \beta_{M^-} = \frac{M^-}{\mathrm{Var}(m)}. \tag{18}$$

Here, the numerators correspond to semicovariances discussed in previous section. The mixed-sign semicovariances (M^+ and M^-) were redefined with a negative sign, to make them positive, for easier interpretation.

$$\beta_{M^+} \equiv -\frac{M^+}{\mathrm{Var}(m)}, \quad \beta_{M^-} \equiv -\frac{M^-}{\mathrm{Var}(m)} \tag{19}$$

Since true betas and semibetas are not observable, Bollerslev et al. (2022) relied on realized semibetas for their study. Where they added semibeta of high frequency returns over a day to calculate semibetas of the day.

The CAPM only considers a single market beta, which is a combination of all semibetas, while the downside version of CAPM considers an upside beta, which combines P and M^+ semibetas, and a downside beta, which combines N and M^- with their individual risk premiums.

Considering four assets each with a traditional CAPM beta equal to 1, since all four assets have the same beta, traditional CAPM will predict similar expected returns for all four assets. Bollerslev et al. (2022) showed that for assets that are highly correlated with the market during upturns and downturns have higher expected excess returns than assets that are highly correlated with the market during upturn and weakly correlated during downturns ($\beta_N < \beta_P$), and assets that are more strongly correlated with market returns during downturn ($\beta_N > \beta_P$) have even higher expected excess returns, while all these assets have the same CAPM beta.

4 Methodology

In this section we will outline the steps taken to analyse the predictive power of semibetas and other factors in asset pricing. We will go through the data preparation techniques, the statistical models employed and the analytical methods used to test whether semibetas provide better predictive power compared to traditional asset pricing models like CAPM and the Fama-French model. This section begin with detailing the data collection and preprocessing, including the cleaning and filtering of stocks returns and illiquidity adjustments. This stage ensures that the dataset is comprehensive, accurate, and suitable for the analysis of stock market behavior over time. The Fama-French factors are integrated into the dataset, along with key financial variables such as returns, price, and volume, spanning several years to provide a robust time series for analysis. Next, the methodology explains how semibetas are calculated for further processing. In addition to semibetas, higher-order risk factors like coskewness and cokurtosis are introduced to capture asymmetry and tail risk in asset returns. These measures are crucial for understanding non-normal return distributions. We will then discuss the regressions used to estimate risk premia for these factors.

4.1 Data Source

The primary data source used in this study is the CRSP (center for Research in Security Prices) database. We will investigate monthly, weekly, daily realised semibetas calculated based on daily stock returns from the period, January 1963 to December 2019. We have chosen this period for the study to ensure a comprehensive and robust dataset that covers multiple market cycles and economic conditions. The extended market period allows for a thorough examination of semibetas in various market environments. By including both bullish and bearish periods, the analysis can better capture the asymmetric relationships between stock returns and market performance, particularly focusing on how these relationships may differ during times of market downturns versus upturns. Ending in 2019 ensures that the dataset covers recent financial phenomena and includes data through the aftermath of the 2008 financial crisis.

4.1.1 Stock data

We extracted the data for following parameters from the CRSP database for the period Jan 1963 to Dec 2019 .

- **PERMNO**: This is a unique identifier assigned to each stock.
- **Date**: The variable date contains the date of stock return observation.
- **SHRCD**: The sharecode carries information regarding the type of stock. We have used sharecode 10 and 11 for our analysis, which represents common shares and non-voting common shares respectively. Common shares are widely traded and liquid, hence are the primary focus in asset pricing studies. This selection will help

in creating dataset of securities with similar characteristics like liquidity, trading volumes and market behaviour

- **PERMCO**: A unique identifier assigned to each company, which is different from PERMNO (used for tracking a company's stock even if the PERMNO changes).
- **PRC**: The closing price of the stock for the given date.
- **VOL**: The trading volume, we use this variable to remove stocks with low trading volumes.
- **RET**: We have also calculated the daily log returns of the stocks and used that in our analysis.
- **DLSTD**: Delisted stock data, to remove delisted stocks
- **SHROUT**: Number of shares outstanding for the stock.

4.1.2 Fama-french data

Following financial indicators have been used for the fama-French data.

- **Date**: The specific date corresponding to the factor data.
- **mktrf**: Market Risk Premium, which reflects the difference between the market return and the risk-free rate.
- **smb**: Size Premium, measuring the return difference between small and large-cap stocks .
- **hml**: Value Premium, indicating the return spread between value stocks (high book-to-market value) and growth stocks (low book-to-market value).
- **umd**: Momentum Factor, representing the tendency for stocks with good past performance to continue outperforming (Up Minus Down momentum).

4.2 Data Cleaning Process

Due to the large size of the CRSP dataset, the data is processed in chunks of 100,000 rows at a time. This chunking process ensures efficient memory usage and speeds up data handling. The data cleaning steps employed aim to standardize, remove outliers, and ensure consistency across the dataset.

- **Date Conversion and Alignment**: The 'Date' column is converted into a standardized datetime format for both the CRSP and Fama-French datasets. This uniformity ensures that both datasets are aligned temporally, enabling accurate merging and comparison across the two data sources.
- **Numeric Parsing**: Several numeric columns, such as price (PRC), volume (VOL), and returns (RET), are parsed to ensure proper numeric formatting. Any non-numeric entries are coerced into NaN (missing) values to ensure the integrity of the dataset.
- **Penny Stock Removal**: To focus on more stable and liquid stocks, the dataset is filtered to remove penny stocks by excluding any stock with a price below $5. This threshold ensures the exclusion of highly volatile, low-value stocks that could distort the analysis.
- **Illiquid Stock Exclusion**: Stocks with low trading volume are considered illiquid and are removed from the dataset. The threshold used for illiquidity is set at the 5th percentile of trading volume, meaning stocks that fall below this value are excluded. This ensures that the analysis focuses on stocks with sufficient market activity to provide reliable data.
- **Handling of Delisted Stocks**: Delisted stocks, which may have gaps in their trading history or missing data, are removed. This ensures that no stock's return or price data is left incomplete, minimizing the risk of errors in subsequent calculations.
- **Forward and Backward Filling**: After handling delisted stocks and missing data, the returns and prices are further refined by forward-filling (using the last known value for subsequent missing values) and backward-filling (using the next available value for preceding missing entries). This process ensures that any gaps in the time series are filled, improving the continuity of the data.
- **Handling Months with Insufficient Data**:For certain stocks, there may be months where insufficient data is available, particularly in cases where a stock was not traded frequently. To address this, any month where a stock traded fewer than 10 days is flagged, and the data for that month is replaced with NaN to avoid skewed results.
- **Winsorization of Data**: Winsorization is applied to reduce the influence of extreme outliers in stock returns. Specifically, the returns are winsorized at the 1st and 99th percentiles, meaning that extreme values outside this range are capped to the nearest percentile value. This prevents outliers from disproportionately affecting the analysis, particularly in regression models where these extreme values could distort results.

- **Removal of Stocks with Incomplete Data**: After applying all the data cleaning steps, any stock that still has missing data for any of the dates is removed from the dataset. This ensures that the final dataset contains only stocks with complete and consistent data across the entire time period. It leaves us with a small dataset of 194 stocks with 56 years of data

By following these steps, the CRSP dataset is refined and prepared for the main analysis. The removal of illiquid stocks, handling of missing data, and winsorization of extreme values ensure that the dataset is robust, accurate, and suitable for conducting the subsequent regression analyses. These procedures allow for the reliable computation of key financial variables such as semibetas, coskewness, and cokurtosis.

4.3 Market Returns

For this study, we developed our own market index. The construction of this value-weighted market index is based on the daily market capitalization of each individual stock sourced from the CRSP database.

- **Market Capitalization**: Market capitalization, often referred to as "market cap," reflects the total market value of a company's outstanding shares. In this study, market capitalization is calculated for each stock by multiplying its Ptice (PRC) to Shares Outstanding (SHROUT). This measure allows us to understand the size of each company relative to others in the dataset. The market capitalization is used in further calculations, for weighting stock returns when calculating overall market returns.

- **Daily Market Returns**:

  Daily market returns represent the overall return of the stock market for a specific day. We calculated these returns by taking a weighted average of the returns of all individual stocks in the dataset, with weights determined by each stock's market capitalization. This method assigns greater influence to larger stocks, ensuring that the market return is accurately represents. Daily market return has been calculated as below .

$$\text{Daily Market Return} = \frac{\sum(\text{Stock Return} \times \text{Market Capitalization})}{\sum \text{Market Capitalization}} \tag{20}$$

In this formula, the individual stock returns are multiplied by their respective market capitalizations to reflect their contribution to the overall market movement. By summing the weighted returns and dividing by the total market capitalization, we obtain the market return for that particular day.

4.4 Beta Calculations

This section describes the process of calculating **semibetas** to examine the relationship between stock returns and market returns under different market conditions. Below is the methodology used to compute semibetas.

4.4.1 Preparation of Returns:

We start by extracting daily stock returns (r_i) and market returns (r_m) from the dataset. Both the stock returns and market returns are aggregated to monthly frequencies for further analysis.

Positive and Negative Return Separation:

- **Positive stock returns** (r_i^+): These represent the positive daily returns of stock i. We construct r_i^+ by setting all negative daily returns to zero.
- **Negative stock returns** (r_i^-): These represent the negative returns of stock i. We set all positive daily returns to zero to calculate r_i^-.

Similarly, we calculate the **positive** (r_m^+) and **negative market returns** (r_m^-) to capture the market's performance during positive and negative phases.

4.4.2 Computation of Semibetas:

We then compute semibetas under different conditions by comparing the stock and market returns[4]. The general formula for semibetas as given by Patton is :

$$\beta_{\text{semi}} = \frac{\sum (r_i \cdot r_m)}{\sum r_m^2} \tag{21}$$

Bollerslev et al. (2022)

where r_m, r_i represent the returns from the market and the stock respectively . The denominator in each case represents the market variance. $\sum$ here represents monthly aggregation.

- **Beta Negative (β_N):** Beta Negative (β_N) measures a stock's sensitivity to negative market returns. It is calculated when both the stock and market returns are negative. The function to calculate beta negative takes the negative stock returns (r_i^-), negative market returns (r_m^-) and total market returns (r_m) as inputs and applies the following formula:

[4]The semibeta calculations used in this paper use the same approach as Bollerslev et al. (2022)

$$\beta_N = \frac{\sum(r_i^- \cdot r_m^-)}{\sum r_m^2} \tag{22}$$

Bollerslev et al. (2022)

In this formula, the numerator captures the product of negative stock and negative market returns, while the denominator reflects the total variance of market returns ($\sum(r_m)^2$).

Beta Positive (β_P): Beta positive (β_P) measures a stock's sensitivity to positive market returns. The function for beta positive uses positive stock returns (r_i^+) and positive market returns (r_m^+) and calculate beta positive as below:

$$\beta_P = \frac{\sum(r_i^+ \cdot r_m^+)}{\sum(r_m)^2} \tag{23}$$

Bollerslev et al. (2022)

This formula is similar to that of beta negative, except that it focuses on positive returns. The numerator captures the product of positive stock and positive market returns, while the denominator represents the total variance of market returns.

Mixed Condition Betas: We also calculate semibetas for situations where stock and market returns move in opposite directions. These are often referred to as mixed-betas:

Mixed Beta (β_{M+}): This beta uses (r_i^-) and (r_m^+) to calculate semibeta using similar calculations as above. It measures the stock's sensitivity when the stock's return is negative during a market upturn:

$$\beta_{M+} = -\frac{\sum(r_i^- \cdot r_m^+)}{\sum(r_m)^2} \tag{24}$$

Bollerslev et al. (2022)

Mixed Beta (β_{M-}): This beta is calculated on (r_i^+) and (r_m^-).

$$\beta_{M-} = -\frac{\sum(r_i^+ \cdot r_m^-)}{\sum(r_m)^2} \tag{25}$$

Bollerslev et al. (2022)

To make the interpretation of risk premium estimates more straightforward in our analysis, a negative sign is added to mixed betas, to make them positive $\beta_{M+} \equiv -\frac{M^+}{\text{Var}(f)}$ and $\beta_{M-} \equiv -\frac{M^-}{\text{Var}(f)}$. We introduce the negative sign to simplify the understanding of these semibetas in the empirical results.

Market Beta (β_{MKT}): In addition to semibetas, we also calculate the overall market beta (β_{MKT}) used by CAPM, to compare our results with the traditional beta. The computation is as below, where we do not consider the signs of the returns, and all returns are included.

$$\beta_{\text{MKT}} = \frac{\sum (r_i \cdot r_m)}{\sum r_m^2} \tag{26}$$

This captures the stock's total exposure to market risk by considering both positive and negative market returns. The numerator measures the covariance, while the denominator represents the variance of market returns.

Functions for Semibeta Calculation: We implemented the semibeta calculation using the following Python functions:

- The `fn_semibeta_cal` function takes signed stock returns, signed market returns, and total market returns as inputs. It calculates the semibetas using the formulas outlined above. The same function is also used to compute the market beta by providing all asset returns (both positive and negative) and total market returns as inputs.
- The `fn_mixed_semib` function calculates the mixed betas by adjusting the signs accordingly when stock and market returns move in opposite directions.

Winsorization: After calculating the semibetas for all stocks and months, we apply winsorization at the first and ninety-nineth percentiles to mitigate the impact of extreme outliers. Winsorization ensures that extreme values do not distort the analysis.

4.5 Comparison Models

In addition to semibetas, this study also employs other asset pricing models in the regression analysis to evaluate their effectiveness and compare them with the semibetas approach. These models are briefly explained in this section to provide context for their role in the comparison.

4.5.1 FAMA-French fators

An improvement over the Capital Asset Pricing Model (CAPM) was proposed by Fama & French (1993) by adding additional fators commonly known as fama french factors. Their model also included factors for size, book value of the firm and momentum of the stock. These factors also incorporate risk taken by an investor by investing in smaller, value stocks that have not been performing well.

- **Size Factor (SMB)**: The "Small Minus Big" (SMB) factor represents the difference in returns between small and large companies. The size of the companies is decided by its market capitalisation. Fama & French (1993) observed that smaller companies tend to outperform larger ones due to their higher growth potential or because they are considered riskier. The SMB factor captures this size effect and is used to calculate the premium that an investor expects for taking the risk of investing in smaller companies.

- **Market Factor (MKT)**: This factor represents the market premium, a premium that an investors expects for investing in the market instead of a risk-free asset. It is the difference between the market return and the risk-free rate. This factor is common between CAPM and Fama-french model.

- **Value Factor (HML)**: "High Minus Low" (HML) reflects the value premium. Stocks that are undervalued have high book to market ratio, they are considered cheap because their market prices are low compared to their intrinsic value, these stocks are called value stocks and stocks with low book to book to market ratios are called growth stocks. The value premium arises because value stocks are often seen as riskier or underpriced, and thus offer higher expected returns compared to growth stocks.

- **Momentum Factor (UMD)**: In addition to the traditional Fama-French three factors, the momentum factor (UMD) was added to the fama french three factor model by Carhart (1997). This factor reflects the tendency for stocks that have shown good performance in the recent past to maintain that momentum in the short term, while stocks with poor performance tend to continue underperforming. The momentum effect has been widely documented, and it highlights the persistence of stock price trends beyond what is explained by size and value.

The Carhart four factor model is commonly represented as Expected return $= \alpha_i + \beta_{\text{mkt}} \times \text{market risk premium} + \beta_{\text{HML}} \times \text{HML} + \beta_{\text{SMB}} \times \text{SMB} + \beta_{\text{UMD}} \times \text{UMD} + \text{error term}$. Where α_i represents the intercept or the stock's return, often referred to as alpha, which cannot be explained by the factors in the model. The coefficients β_{mkt}, β_{HML}, β_{SMB}, and β_{UMD} represent the stock's sensitivity to the respective factors. The variables HML, SMB, UMD represent the values of the factors on a particular day,

These factors are derived from empirical research and have been shown to significantly improve the explanatory power of models that predict stock returns. Each factor represents a systematic source of risk that influences stock performance. In this study, we integrate the Fama-French factors (extracted from CRSP database from Jan 1963 to Dec 2019) with the stock data obtained from the CRSP database. The factors are merged with the CRSP data by matching dates to ensure consistency.

4.5.2 Coskewness and Cokurtosis

With Semibetas we aim to capture non-Normally distributed systematic risks. In addition to semibetas, researchers have explored other statistical measures to capture the asymmetric relationships between stock and market returns. One of the most widely studied measures is coskewness, introduced by Kraus & Litzenberger (1976) . Similarly, research by Dittmar (2002) suggests that cokurtosis may also be an important factor in pricing stocks. We have calculated these factors on monthly basis to explain monthly returns from the stocks.

Let:

$$r_{\text{diff}} = \text{daily asset return} - \text{monthly mean of asset return}$$

$$m_{\text{diff}} = \text{daily market return} - \text{monthly mean of market return}$$

The monthly coskewness can then be written as:

$$\text{Monthly Coskewness} = \frac{\frac{1}{d}\sum(r_{\text{diff}})(m_{\text{diff}}^2)}{\sqrt{\left(\frac{1}{d}\sum(r_{\text{diff}}^2)\right)\left(\frac{1}{d}\sum(m_{\text{diff}}^2)\right)}}$$

The monthly cokurtosis can then be written as:

$$\text{Monthly Cokurtosis} = \frac{\frac{1}{d}\sum(r_{\text{diff}})(m_{\text{diff}}^3)}{\sqrt{\left(\frac{1}{d}\sum(r_{\text{diff}}^2)\right)\left(\frac{1}{d}\sum(m_{\text{diff}}^2)\right)^3}}$$

where d represents the number of trading days in the month.

In practice, we implement these formulas using Python functions, which resample the data to a monthly frequency, compute the deviations from the mean, and then calculate coskewness and cokurtosis based on the resampled data.

4.5.3 Upside and downside Beta

In addition to the standard semibetas, we also calculate **beta up** and **beta down**, as proposed by Ang et al. (2006). These betas distinguish between market conditions with positive and negative returns. Specifically:

Upside Beta ($\hat{\beta}^+$) measures the stock's sensitivity to positive market returns and monthly upside beta is computed as[5]:

$$\hat{\beta}^+_{t,i} = \frac{\sum_{k=1}^{d} r_{t,k,i} \cdot f^+_{t,k}}{\sum_{k=1}^{d} (f^+_{t,k})^2} \tag{27}$$

Where $r_{t,k,i}$ is the return of stock i on day k of month t, and $f^+_{t,k}$ is the market return

[5] formulas for upside and downside betas consistent with Bollerslev et al. (2022)

on day k, conditioned on the market having a positive return.

Downside Beta ($\hat{\beta}^-$) measures the stock's sensitivity to negative market returns and monthly downside beta is computed as:

$$\hat{\beta}_{t,i}^- = \frac{\sum_{k=1}^{d} r_{t,k,i} \cdot f_{t,k}^-}{\sum_{k=1}^{d} (f_{t,k}^-)^2} \tag{28}$$

Where $f_{t,k}^-$ represents the market return on day k, when the market experiences a negative return. These betas provide insights into how stocks perform during market upturns and downturns.

4.6 Regression Models

We used linear regression approach for estimating the risk premia. We iterated through each monthly period to compute CAPM coefficients based on monthly returns. The Newey-West correction was applied to the standard errors to account for potential autocorrelation and heteroscedasticity in the data. The regression results present the estimated coefficients and t-statistics for each variable included in the analysis.

Our analysis starts with results from cross-sectional predictive regressions. These regressions enable us to estimate the risk premiums, denoted by lamdas, for each semibeta simultaneously.We conduct a cross-sectional regression for each month $t+1$ by using data from all stocks i = 1 to N that are present in both the current month t and the subsequent month $t+1$.

$$\lambda_{0,t+1} + \lambda_{1,t+1}\hat{\beta}_{t,i}^{M+} + \lambda_{2,t+1}\hat{\beta}_{t,i}^{N} + \lambda_{3,t+1}\hat{\beta}_{t,i}^{P} + \lambda_{4,t+1}\hat{\beta}_{t,i}^{M-} + \epsilon_{t+1,i} = r_{t+1,i} \tag{29}$$

In this regression, $r_{t+1,i}$ is the return of stock i in month $t+1$, and $\hat{\beta}_{t,i}^{N}$, $\hat{\beta}_{t,i}^{P}$, $\hat{\beta}_{t,i}^{M+}$, and $\hat{\beta}_{t,i}^{M-}$ are the estimated downside, upside, and mixed semibetas, respectively. The corresponding coeffiencts (lambdas) represent the risk premiums for each semibeta, while $\epsilon_{t+1,i}$ is the error term.

Following the estimation of risk premiums, we compute the average risk premiums for each semibeta by taking the averages of the λ's across all months:

These time-series averages, with their average Newey-West t-stat, and average of the R^2's, are reported as key results. The lags for newey west t-stat are calculated as lags = 0.75 × number of monthly observations$^{1/3}$. This calculation of lags is often used in financial econometrics.

4.6.1 CAPM Regression:

We first run a simple linear regression with CAPM beta. In this model, monthly stock returns are regressed against the traditional realised market beta to assess each stock's sensitivity to market movements. Here, the dependent variable is the monthly return_{t+1}, and the independent variable is the market beta_t.

4.6.2 Semibeta Regressions:

Following the CAPM regression, we performed predictive regressions between semibetas and stock returns.

- **Dependent variable**: Monthly stock returns $r_{t+1,i}$
- **Independent variables(Semibetas)**: $\hat{\beta}^{N}_{t,i}$, $\hat{\beta}^{P}_{t,i}$, $\hat{\beta}^{M+}_{t,i}$, $\hat{\beta}^{M-}_{t,i}$

These regressions enable us to estimate the individual contribution of downside, upside, and mixed semibetas to stock returns and assess their predictive power across different market conditions.

4.6.3 Fama-French and Semibeta Regressions:

Next we incorporate the Fama-French three factors, size (SMB), value (HML), and momentum (UMD), into the model to examine how these factors, combined with semibetas, explain stock returns. The extended regression model is:

- **Dependent variable**: Monthly stock returns $r_{t+1,i}$
- **Independent variables (Fama factors and Semibetas)**: $\hat{\beta}^{N}_{t,i}$, $\hat{\beta}^{P}_{t,i}$, $\hat{\beta}^{M+}_{t,i}$, $\hat{\beta}^{M-}_{t,i}$, SMB_t, HML_t, UMD_t

4.6.4 Coskewness and Cokurtosis Regressions:

We then incorporate higher-order moments, coskewness and cokurtosis, into the regression models. Coskewness and cokurtosis capture the asymmetry in stock and market return distributions . The regression model has:

- **Dependent variable**: Monthly stock returns $r_{t+1,i}$
- **Independent variables**: $\text{coskewness}_{t,i}$, $\text{cokurtosis}_{t,i}$

4.6.5 Upside and Downside Beta Regressions

In this regression, monthly stock returns are regressed against upside and downside betas. These betas, first introduced by Ang et al. (2006), capture the stock's sensitivity to market movements during positive and negative market conditions, respectively. While upside and downside betas measure how a stock's returns respond to positive and negative market movements, semibetas also consider the signs of asset returns in these market movements.

Beta Up ($\hat{\beta}^{+}_{t,i}$) captures covariance of stock with positive market returns only, while Beta Down ($\hat{\beta}^{-}_{t,i}$) captures covariance of stock with negative market returns only.

- **Dependent variable**: Monthly stock returns $r_{t+1,i}$
- **Independent variables(upside, downside beta)**: $\hat{\beta}^{+}_{t,i}$, $\hat{\beta}^{-}_{t,i}$

4.6.6 Other Regressions:

In addition to the regressions mentioned earlier, we performed further analyses that combine factors from the previous regressions. For instance, in the following regression model, we include β^{+}, β^{-}, and the semibetas β^{N}, β^{P}, β^{M+}, and β^{M-} to examine their combined effects on stock returns.

The regression equation for this model is given as:

- **Dependent variable**: Monthly stock returns $r_{t+1,i}$
- **Independent variables(upside, downside beta and semibetas)**: $\hat{\beta}^{N}_{t,i}$, $\hat{\beta}^{P}_{t,i}$, $\hat{\beta}^{+}_{t,i}$, $\hat{\beta}^{-}_{t,i}$, $\hat{\beta}^{M+}_{t,i}$, $\hat{\beta}^{M-}_{t,i}$

We also performed regressions by including higher-order moments such as coskewness and cokurtosis along with the semibetas.

- **Dependent variable**: Monthly stock returns $r_{t+1,i}$
- **Independent variables(upside, downside beta and semibetas)**: $\hat{\beta}^{N}_{t,i}$, $\hat{\beta}^{P}_{t,i}$, $\hat{\beta}^{M+}_{t,i}$, $\hat{\beta}^{M-}_{t,i}$,$\text{coskewness}_{t,i}$, $\text{cokurtosis}_{t,i}$

5 Results

In this section, we present the findings of our analysis, focusing on how various semibetas capture the sensitivity of stock returns to different market conditions. Using daily CRSP stock data spanning from January 1963 to December 2019, we estimate semibetas to assess how stocks react to both concordant (moving in the same direction as the market) and discordant (moving in opposite directions) movements. We start by examining the summary statistics of semibetas and correlations between betas. This is accompanied by a series of regression analyses. The results highlight that semibetas provide significant explanatory power beyond what the traditional CAPM can offer.

5.1 Monthly Returns

In this section, we present our regression results on monthly returns to assess the role of semibetas in explaining stock return behavior. We begin by summarizing the data and key statistics of semibeta estimates, comparing them to traditional beta measure.

5.1.1 summary statistics

Table1 provides the summary statistics of the monthly semibeta estimates, based on average of summary statistics of all the stocks in the sample. These averages are calculated for each time period (for a particular month) for all stocks and then averaged over the entire time series.

From the results in table we can see that β_P and β_N, also referred to as concordant semibetas are larger than the discordant semibetas (β_{M+} and β_{M-}) which implies that on average, stocks and markets move in the same direction, both postive (or negative), more than they move in the opposite directions. The market beta was found to be the highest with mean value of .8182 indicating that, on average, the stocks in the sample exhibit a less-than-proportional response to market movements (as a beta of 1 would indicate a proportional response)

	Mean	Median	Std Dev
B	0.818242	0.738786	0.765144
B_N	0.549143	0.466936	0.483576
B_P	0.598209	0.542774	0.312438
B_{M+}	0.176186	0.133696	0.175892
B_{M-}	0.152924	0.115693	0.142541

Table 1: Monthly Realised Semibeta Summary Statistics

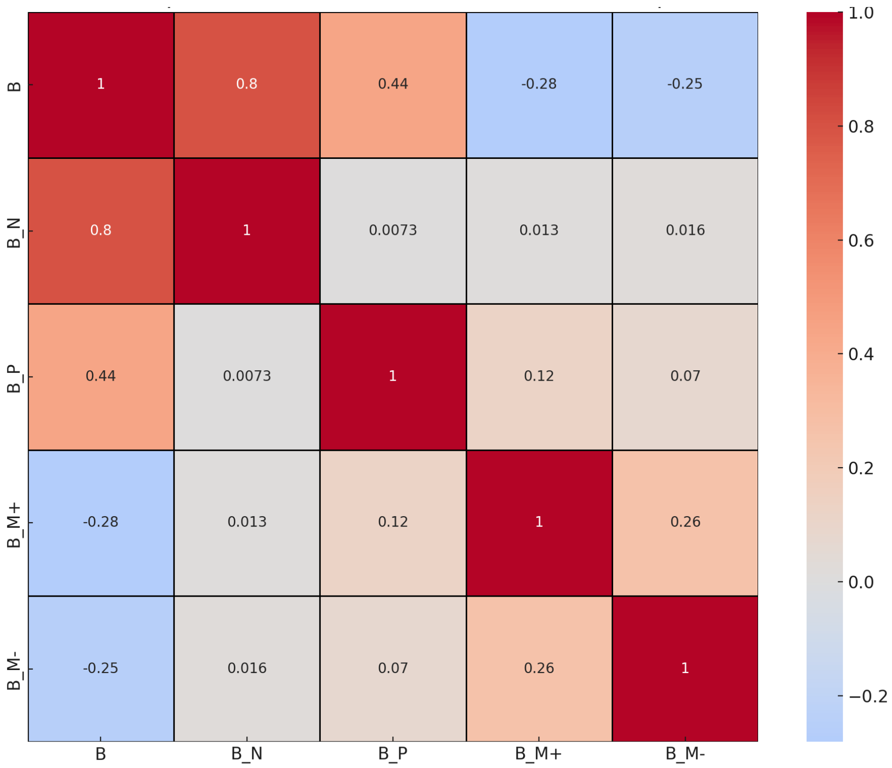

Figure 4: Correaltion heatmap of monthly Realised semibetas

figure-4 presents averages for the correlations between betas. It shows that the correlation between concordant betas(β_P and β_N) and market beta is higher than discordant betas but the correlation is less than one which implies that semibetas may convey some additional information which has not already been covered by market beta.

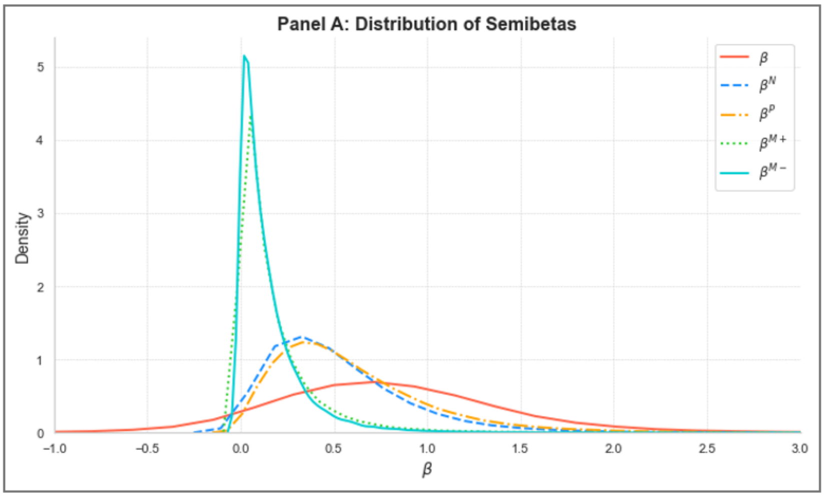

Figure 5: Unconditional distribution of monthly semibetas

Figure 5 shows differences between the semibetas and traditional betas, the figure presents the unconditional distributions of these betas across all the observations in our sample. The distribution of the traditional beta (β) exhibits a slightly symmetric shape and is centered close to 1. However, the distributions of the realized semibetas (β_N, β_P, β_{M+}, and β_{M-}) are noticeably right-skewed.

As seen from the figure, the concordant semibetas (β_P and β_N) have distributions that are very similar. Similarly, the discordant semibetas (β_{M+} and β_{M-}), which reflect cases where the stock and market move in opposite directions, also exhibit close distributions. All semibetas have mean value less than 1, consistent with the summary statistics provided earlier.

5.1.2 Regression Results

We start our examination by applying the Fama & MacBeth (1973) approach to cross-sectional regressions. This method enables us to determine the risk premiums (λ) associated with each semibeta simultaneously. We compute the risk premiums for semibetas in month $t+1$ using the following regression model:

$$\lambda_{0,t+1} + \lambda_{t+1}^{M+}\hat{\beta}_{t,i}^{M+} + \lambda_{t+1}^{M-}\hat{\beta}_{t,i}^{M-} + \lambda_{t+1}^{N}\hat{\beta}_{t,i}^{N} + \lambda_{t+1}^{P}\hat{\beta}_{t,i}^{P} + \epsilon_{t+1,i} = r_{t+1,i} \tag{30}$$

Once we have obtained these cross-sectional estimates of parameters for $T-1$ periods, the average risk premiums are calculated by averaging the λ values over the entire timeline for each semibeta.

We first calculate the risk premium for the traditional CAPM. Our results in table 2, table 3 indicate that the traditional market beta is significant in explaining monthly returns and explains 10.7 percent variation of monthly returns, while semibetas provide additional insights. The R-squared increased from 10.7 percent to 61.4 percent when we included semibetas. These findings align with the notion that traditional beta alone is insufficient to capture the full complexity of stock sensitivities to market fluctuations. We also tested the statistical significance of the increase in R square, to check if the semibeta-based model provides more useful information than the traditional CAPM model, we check if the semibeta model reduces to CAPM i.e $H_0 : \lambda_N = \lambda_P = -\lambda_{M+} = -\lambda_{M-}$. we conducted wald test to test this and reject the null hypothesis.

The semibetas demonstrate strong statistical significance, with all t-statistics well above conventional thresholds and extremely low p-values (all < 0.001). This suggests that these semibetas play an important role in explaining variations in stock returns. The results indicate that β_N and β_P are the most significant factors, with newy-west t-statistics of 24.41 and 16.12, respectively. Both have positive coefficients (0.09808 for β_N and 0.09952 for β_P), suggesting that they predict higher future returns. These high coefficients imply that stocks moving in the same direction as the market either positively or negatively tend to generate higher returns. β_{M+} and β_{M-} are also significant with coefficients of opposite signs, where coefficient of β_{M-} is negative predicting lower future returns.

We conducted a hypothesis test using the Two-Sample T-Test ,for equality of means, to examine whether there is asymmetry in how returns respond to positive and negative market movements. $H_0 : \lambda_P = \lambda_N$ and $\lambda_{M+} = \lambda_{M-}$. The results showed statistically significant differences in both cases, with very low p values. Which confirms that the market does not respond symmetrically to positive and negative movements, supporting the idea that downside risks have different implications than upside risks in asset pricing.

	CAPM
Coefficient(%)	2.63714
t-Statistic	6.97763
R-squared (%)	10.79882

Table 2: Averaged regression results for regression with monthly asset returns as dependent and monthly CAPM beta as independent variable

	β_N	β_P	β_{M+}	β_{M-}
Coefficient(%)	0.09808	0.09952	0.13455	-0.13205
t-Statistic	24.40716	16.11565	12.23817	9.47941
R-squared (%)	61.422			

Table 3: Averaged regression results for regression with monthly asset returns as dependent and monthly semibetas as independent variable

Literature has identified several other factors that affect asset returns, as discussed in the previous section. To further explore this, we test the significance of these factors in explaining stock returns along with semibetas.

We start with the fama and Carhart factors of size(SMB), book-to-market ratio (HML), and momentum (UMD). Table 4 shows the results of regression with fama factors and semibetas.

From the results, we observe that including fama factors has not made any substantial change in R square, which indicate that there no additional information contributed by fama factors, all three fama factors represented by SMB, HML, and UMD display low t stat value indicating low significance.On the other hand, the semibetas, particularly β_N and β_P , remain significant, with t-values of 4.13 and 5.14, respectively.

Variable	Coefficient (%)	t-Statistic
β_N	0.56198	4.13509
β_P	0.37758	5.14795
β_{M+}	0.30133	2.07735
β_{M-}	-0.22858	3.16911
smb	0.04339	0.41451
hml	0.00453	0.07120
umd	-0.03794	0.76174
R-squared (%)	62.36740	

Table 4: Averaged regression results for regression with monthly asset returns as dependent and Fama French and semibetas as independent variable

Next, we consider the upside and downside betas as proposed by Ang et al. (2006). While the semibetas calculate four distinct covariations based on the signs of the market and asset returns, the upside and downside betas decompose traditional beta into two components—one for market upturns and one for downturns.

Table 5 presents the results for the upside and downside betas alone, while table 6 includes the results of the regression combining upside and downside betas with the semibetas.

Our findings indicate that the risk premiums for both the upside and downside betas are statistically significant when examined independently. Specifically, in the first regression, the downside beta (β^-) shows a coefficient of .0422 with very high t statistics, while the upside beta (β^+) has a coefficient of -.051960 with highly significant t statistics. These results suggest that stocks are sensitive to market upturns and downturns, with downside risk associated with higher expected returns, consistent with the literature.However, when we include both the semibetas and the upside/downside betas in the same regression, they do not retain statistical significance. As shown in table 6, all semibetas are significant however upside and downside betas are not significant. The R squared increases from 45.18 percent to 62 percent on inclusion of semibetas. This lack of significance may be due to multicollinearity or computational complexity. The high correlation between these measures might interfere with their ability to provide independent explanatory power in the model.

Variable	Coefficient (%)	t-Statistic
β^+	-0.0501960	14.59532
β^-	0.0422090	19.67863
R-squared (%)	45.1889	

Table 5: Averaged regression results for regression with monthly asset returns as dependent and upside and downside betas as independent variable

Variable	Coefficient (%)	t-Statistic
β^{+}	-1.46870	2.68858
β^{-}	0.76920	2.21713
β^{N}	8.93220	11.16149
β^{P}	8.25170	8.25128
β^{M+}	9.86680	5.89179
β^{M-}	-10.72100	6.18992
R-squared (%)	62.96420	

Table 6: Averaged regression results for regression with monthly asset returns as dependent and upside beta, downside beta and semibetas as independent variable

In addition to semibetas, coskewness and cokurtosis also account for non-Normal, asymmetric dependencies in asset returns. These measures capture higher-order moments, with Coskewness capturing the asymmetry in returns, while cokurtosis represents 'fat tails', indicating the likelihood of extreme outcomes relative to a normal distribution. To assess the significance of these factors in explaining monthly returns, we computed monthly coskewness and cokurtosis and incorporated them into our regression models. Table 7 and table 8 shows the results of our regressions. In the first model, both coskewness and cokurtosis were found to be statistically insignificant. This suggests that on their own, these higher-order moments do not have significant explanatory power for monthly stock returns. However, when coskewness and cokurtosis are included alongside semibetas in the second regression, coskewness becomes statistically significant with a t-statistic of 10.38. All semibetas were also found to be significant, with β_N and β_P being the most significant. β_N had a t-statistic of 27.86 , while β_P had a t-statistic of 19.45 . The inclusion of semibetas, coskewness, and cokurtosis improves the explanatory power of the model substantially. The R-squared value increased from 10.7 percent in the model with only market beta as the sole factor to 67.08 percent when semibetas and other factors were included, indicating that semibetas provide a better explanation of return variation.

Variable	Coefficient (%)	t-Statistic
$coskewness$	0.21820	0.79424
$cokurtosis$	0.03630	0.85114
R-squared (%)	0.29750	

Table 7: Averaged regression results for regression with monthly asset returns as dependent and coskewness and cokurtosis as independent variable

Variable	Coefficient (%)	t-Statistic
Coskewness	-0.01801	-10.38555
Cokurtosis	-0.00035	-1.03014
β_N	0.10765	-27.86399
β_P	0.11460	19.45719
β_{M+}	0.15186	-14.29385
β_{M-}	0.15635	11.53970
R-squared (%)	67.08410	

Table 8: Averaged regression results for regression with monthly asset returns as dependent and coskewness , cokurtosis and semibetas as independent variable

Figure 6 presents the absolute t-statistics of various factors across multiple regressions, allowing us to assess the significance of each factor in explaining asset returns. The horizontal red dashed line marks the 5 percent significance threshold, with any bars above this line indicating statistical significance at the 5 percent level. In particular, we observe that in Regression 2, all semibetas are highly significant, showing t-statistics well above the 5 percent significance line. This confirms that these semibetas play a crucial role in explaining stock returns. Moreover, the significance of the semibetas is consistently high across multiple regressions. Overall, semibetas consistently outperform traditional factors such as market beta in explaining the variation in stock returns across the regressions presented in the figure.

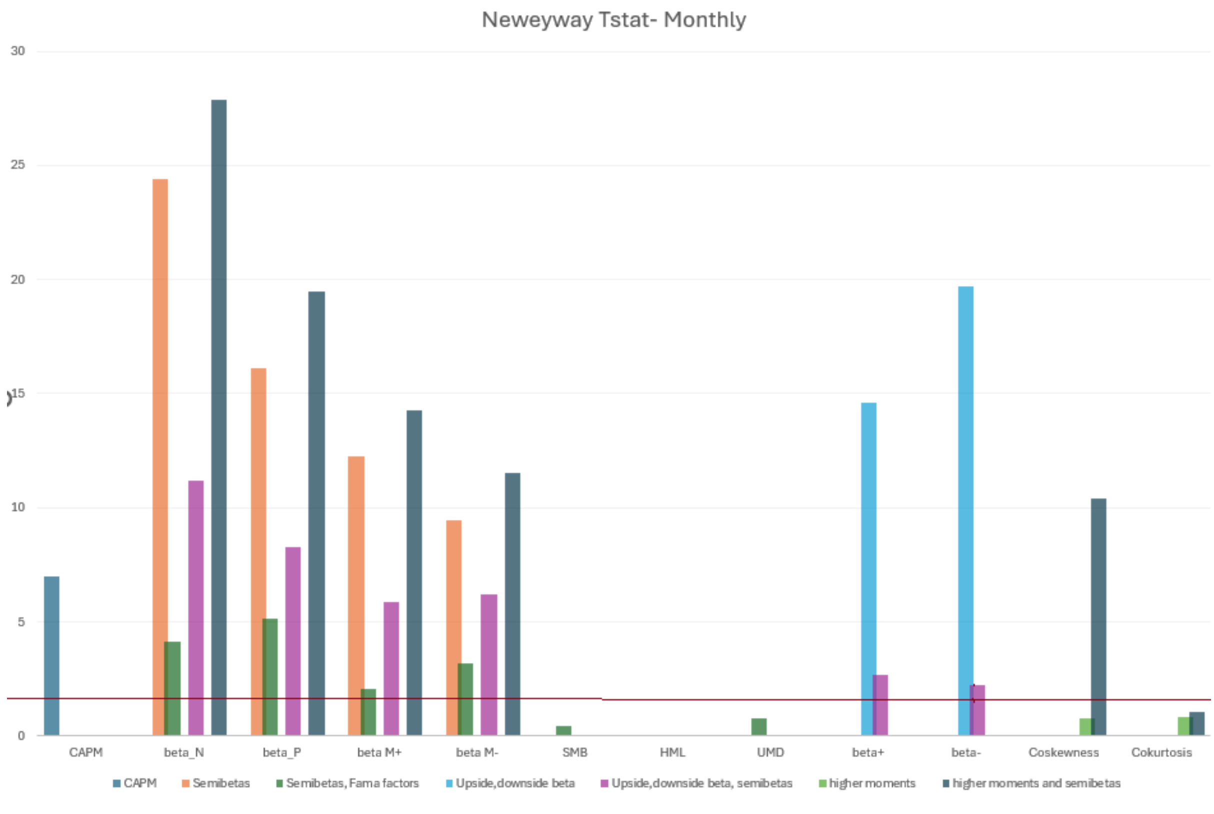

Figure 6: Newey West t-stat for Regressions on Monthly Returns

Table 9 shows the average adjusted R-squared values for various models. From the table, we see that semibetas significantly enhance the efficiency of asset pricing models. The adjusted R-squared value rises substantially from a baseline of 3.68% in the CAPM model to 52.36% when semibetas are included. The inclusion of Fama-French factors and higher moments with semibetas individually further improves adj r square to 56.98% and 62.23% respectively. This increase in the adjusted R-squared indicates that the addition of semibetas enhances the model's ability to explain the variations in asset returns.

Model	**Average Adjusted R-squared (%)**
CAPM	3.68
Semibetas	52.36
Semibetas + Fama-French	56.98
Beta Up and Down	31.62
Up and Down with Semibetas	54.53
Coskewness and Cokurtosis	0.03
Coskewness, Cokurtosis with Semibetas	62.23

Table 9: Average Adjusted R-squared values for different models

5.2 Weekly Returns

In this section, we analyze the significance of semibetas in explaining weekly returns. Similar to the monthly analysis, we assess how semibetas, alongside other factors, contribute in explaining weekly returns. The results are obtained by modifying the code, shared at the end of the dissertation, to resample data on weekly frequency instead of monthly.

Variable	Model 1		Model 2		Model 3	
	Coeff (%)	t-stat	Coeff (%)	t-stat	Coeff (%)	t-stat
CAPM	0.505967	5.61668				
β_N			0.023971	46.8795	0.5430	4.1425
β_P			0.020016	28.9507	0.3392	6.9812
β_M^+			0.020155	21.3006	0.2517	2.0263
β_M^-			-0.019232	16.6356	-0.1961	3.3236
SMB					0.0451	0.0063
HML					1.2349	0.2716
UMD					4.2205	1.7045
R^2 (%)	6.2466		58.3142		58.8147	

Table 10: Regression results using weekly parameters - Part 1

Variable	Model 4		Model 5		Model 6		Model 7	
	Coeff (%)	t-stat	Coeff (%)	t-stat	Coeff (%)	t-stat	Coeff (%)	t-stat
β_N			0.023539	44.4626			0.024418	48.1927
β_P			0.018969	26.2193			0.020751	30.2408
β_M^+			0.019637	19.5086			0.020873	22.1872
β_M^-			0.018804	15.7712			0.02	17.3890
β^+	0.0941	9.5654	-0.0403	6.7374				
β^-	0.0234	6.2362	0.0104	4.3586				
Coskewness					0.0001	0.9215	0.000617	9.3216
Cokurtosis					-0.000031	-0.6313	-0.000041	1.3361
R^2 (%)	5.1745		59.5888		0.0979		59.5356	

Table 11: Regression results using weekly parameters - Part 2

Table 10 and 11 show our results. Starting with the traditional CAPM model we see that the coefficient for the market beta (β_{CAPM}) is 0.5059 with a t-statistic of 5.62, which is statistically significant at conventional levels. The R-squared value is only 6.25 percent which has dropped from (10.7 percent) in case of monthly returns. This shows that the market beta is less significant in explaining returns in shorter duration. when semibetas are introduced in the next model, the R-squared value jumps to 58.31 percent. All semibetas show high significance. Similar to the case of monthly returns, the Fama factors do not add any significant information to the model and are insignificant in explaining weekly returns.The inclusion of upside and downside betas reveals that both β^+ (with a t-statistic of 9.57) and β^- (with a t-statistic of 6.24) are highly significant in explaining weekly returns. However, these upside and downside betas do not appear to add as much explanatory power as semibetas. The R-squared value increases only slightly from 58.3% in model with semibetas alone to 59.59% in model with semibetas and upside,downside betas, demonstrating that semibetas capture much of the variation in weekly returns.In the next model, coskewness and cokurtosis were included, but their t-statistics (0.92 for coskewness and -0.63 for cokurtosis) indicate that they are not statistically significant. However, when combined with semibetas, coskewness becomes significant with a t-statistic of 9.32.

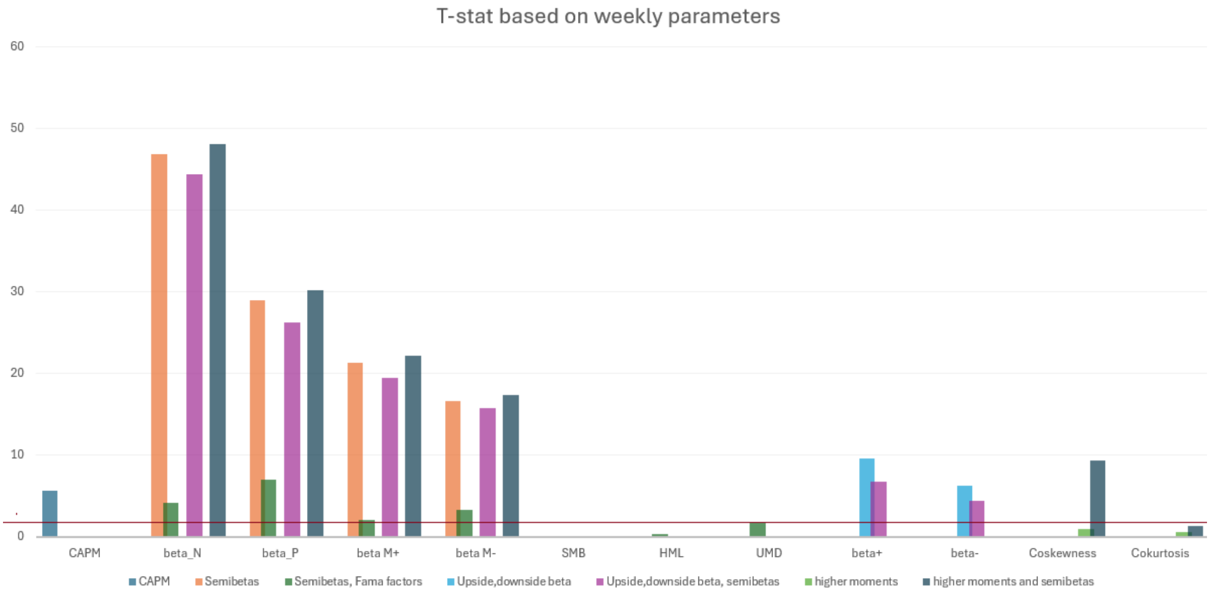

Figure 7: Newey West t-stat for Regressions on Weekly Returns

Figure 7 presents the absolute t-statistics of various factors across multiple regressions. Similar to the case of monthly returns, we observe that all semibetas are highly significant, showing t-statistics well above the 5 percent significance line. This confirms that these semibetas play a crucial role in explaining weekly stock returns.

In both the monthly and weekly return analyses, we observe that β_N (the concordant negative semibeta) consistently exhibits the highest significance across all models and have a positive coefficient, indicating that stocks moving in the same negative direction as the market tend to generate higher future returns.

These findings align with the insights provided by Bollerslev et al. (2022), who emphasized the importance of accounting for downside risk in asset pricing models. The higher predictive power of β_N suggests that investors demand a premium for assets that exhibit stronger co-movements with the market during downturns, highlighting the importance of incorporating semibetas into return prediction models.

5.3 Daily Returns

We also conducted our analysis using daily data from January 2009 to December 2019, derived from the original dataset used in the previous analyses. The regression results on daily returns reveal that semibetas continue to play a significant role in explaining returns, although the explanatory power is reduced compared to longer time horizons. We find that upside and downside betas are the most significant in explaining daily returns

Variable	Model 1		Model 2		Model 3	
	Coeff (%)	t-stat	Coeff (%)	t-stat	Coeff (%)	t-stat
CAPM	0.000253	2.908871				
β_N			0.000209	8.510157	13.66057	4.963601
β_P			0.000142	6.879857	10.7114	6.704766
β_M^+			0.000103	5.839026	10.04203	4.308502
β_M^-			-0.000072	3.512145	-5.75728	4.472063
SMB					24.88627	5.44841
HML					20.27762	4.801593
UMD					-4.6842	2.208433
R^2 (%)	0.454175		0.072		20.93188	

Table 12: Regression results using daily parameters - Part 1

Variable	Model 4		Model 5	
	Coeff (%)	t-stat	Coeff (%)	t-stat
β_N			0.000115	6.028236
β_P			0.000060	3.38341
β_M^+			0.000022	0.829891
β_M^-			-0.000022	3.089841
β^+	0.000084	7.788742	-0.000082	9.113142
β^-	0.000067	6.054763	0.000094	9.188193
R^2 (%)	0.040694		0.072	

Table 13: Regression results using daily parameters - Part 2

Table 12, and table 13 show our result. Overall, the analysis of daily returns confirms that semibetas maintain their significance even at the daily frequency, suggesting their robustness in capturing stock return behavior across different time horizons. We also find that among all the semibetas Beta N remains the most crucial as in previous analyses.

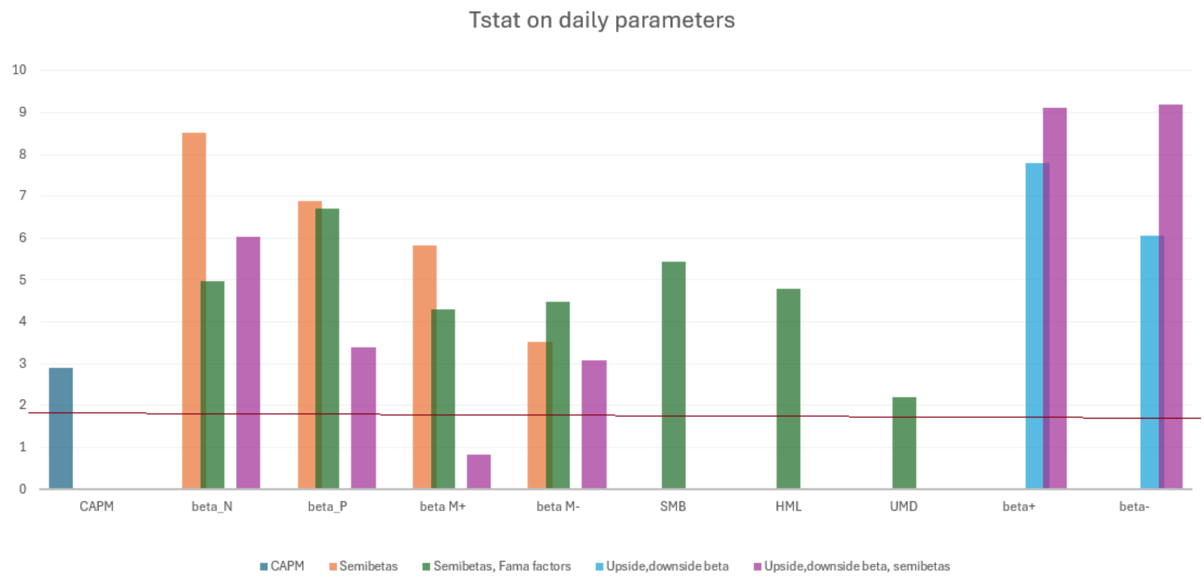

Figure 8: Newey West t-stat for Regressions on Daily Returns

6 Conclusion

In this study we found that that semibetas provide a clearer and more detailed explanation of asset returns. The results indicated that semibetas, particularly those linked to negative market and asset movements, are most significant and predict higher future returns. This confirms that downside risk plays a crucial role in asset pricing, which the CAPM does not fully capture.

When compared with other established models, such as the Fama-French factors, and higher-order factors like coskewness and cokurtosis, semibetas consistently showed greater predictive power.

This study found that semibetas offer a more comprehensive approach to asset pricing by accounting for asymmetric risks and investors can leverage semibetas to assess and manage downside risks more effectively, leading to more robust and diversified portfolios, especially during periods of market stress.

References

Ang, A., Chen, J. & Xing, Y. (2006), 'Downside risk', *The review of financial studies* **19**(4), 1191–1239.

Bhandari, L. C. (1988), 'Debt/equity ratio and expected common stock returns: Empirical evidence', *The Journal of Finance* **43**, 507–528.

Bollerslev, T., Li, J., Patton, A. J. & Quaedvlieg, R. (2020), 'Realized semicovariances', *Econometrica* **88**, 1515–1551.

Bollerslev, T., Patton, A. J. & Quaedvlieg, R. (2022), 'Realized semibetas: Disentangling "good" and "bad" downside risks', *Journal of Financial Economics* **144**, 227–246.

Carhart, M. M. (1997), 'On persistence in mutual fund performance', *The Journal of finance* **52**(1), 57–82.

De Bondt, W. F. & Thaler, R. (1985), 'Does the stock market overreact?', *The Journal of finance* **40**(3), 793–805.

Dittmar, R. F. (2002), 'Nonlinear pricing kernels, kurtosis preference, and evidence from the cross section of equity returns', *The Journal of Finance* **57**(1), 369–403.

Fama, E. F. (1965), 'The behavior of stock-market prices', *The Journal of Business* **38**(1), 34–105.
URL: *http://www.jstor.org/stable/2350752*

Fama, E. F., Fisher, L., Jensen, M. C. & Roll, R. (1969), 'The adjustment of stock prices to new information', *International economic review* **10**(1), 1–21.

Fama, E. F. & French, K. R. (1992), 'The cross-section of expected stock returns', *The Journal of Finance* **47**, 427–465.

Fama, E. F. & French, K. R. (1993), 'Common risk factors in the returns on stocks and bonds', *Journal of financial economics* **33**(1), 3–56.

Fama, E. F. & MacBeth, J. D. (1973), 'Risk, return, and equilibrium: Empirical tests', *Journal of political economy* **81**(3), 607–636.

Kahneman, D. & Tversky, A. (1979), 'Prospect theory: An analysis of decision under risk', *Econometrica* **47**, 263.

Kraus, A. & Litzenberger, R. H. (1976), 'Skewness preference and the valuation of risk assets', *The Journal of finance* **31**(4), 1085–1100.

Lintner, J. (1965), 'The valuation of risk assets and the selection of risky investments in stock portfolios and capital budgets', *The Review of Economics and Statistics* **47**, 13.

Longin, F. & Solnik, B. (2001), 'Extreme correlation of international equity markets', *The Journal of Finance* **56**, 649–676.

Mandelbrot, B. (1963), 'The variation of certain speculative prices', *The Journal of Business* **36**(4), 394–419.
URL: *http://www.jstor.org/stable/2350970*

Mandelbrot, B. (1972), 'Certain speculative prices (1963)', *The Journal of Business* **45**(4), 542–543.

Markowitz, H. (1952), 'Portfolio selection', *The Journal of Finance* **7**, 77.

Merton, R. C. (1973), 'An intertemporal capital asset pricing model', *Econometrica: Journal of the Econometric Society* pp. 867–887.

Merton, R. C. et al. (1987), 'A simple model of capital market equilibrium with incomplete information'.

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

Mossin, J. (1966), 'Equilibrium in a capital asset market', *Econometrica: Journal of the econometric society* pp. 768–783.

Patton, A. J. (2004), ‘On the out-of-sample importance of skewness and asymmetric dependence for asset allocation’, *Journal of Financial Econometrics* **2**, 130–168.

Pontiff, J. (1996), ‘Costly arbitrage: Evidence from closed-end funds’, *The Quarterly Journal of Economics* **111**(4), 1135–1151.

Roll, R. (1977), ‘A critique of the asset pricing theory’s tests part i: On past and potential testability of the theory’, *Journal of Financial Economics* **4**(2), 129–176.
URL: *https://www.sciencedirect.com/science/article/pii/0304405X77900095*

Sharpe, W. F. (1964), ‘Capital asset prices: A theory of market equilibrium under conditions of risk’, *The Journal of Finance* **19**, 425.

7 Code

```


# importing libraries

from scipy.stats import chi2
import patsy
import seaborn as sns
import matplotlib.pyplot as plt
from statsmodels.stats.outliers_influence import
    variance_inflation_factor
import pandas as pd
import numpy as np
import statsmodels.api as sm
from scipy.stats import norm
from scipy.stats.mstats import winsorize
from statsmodels.stats.sandwich_covariance import cov_hac
from sklearn.preprocessing import StandardScaler
from scipy.stats import zscore

# function to remove months with less trading data


def fn_monthoutlier(dailyreturns_data):
    for share_1 in dailyreturns_data.columns:
        tr_days_month = dailyreturns_data[share_1].resample('ME').count
            ()
        less_tr_mon = tr_days_month[tr_days_month < 10].dropna().index
        dailyreturns_data.loc[dailyreturns_data.index.to_period('M').
            isin(
```

```
            less_tr_mon.to_period('M')), share_1] = np.nan
    return dailyreturns_data


# Semibeta calculation- calculates monthly semibeta
def fn_semibeta_cal(signed_stock_returns, signed_market_returns, all_market_returns):
    numerator_df = signed_stock_returns * signed_market_returns.values
    denomerator_df = all_market_returns ** 2
    sum_numerator_df = numerator_df.resample('ME').sum(min_count=1)
    sum_denomerator_df = denomerator_df.resample('ME').sum(min_count=1)
    motnhly_semib_df = sum_numerator_df / sum_denomerator_df.values
    return motnhly_semib_df


def fn_mixed_semib(signed_stock_returns, signed_market_returns, all_market_returns):
    numerator_df = signed_stock_returns * signed_market_returns.values
    denomerator_df = all_market_returns ** 2
    sum_numerator_df = numerator_df.resample('ME').sum(min_count=1)
    sum_denomerator_df = denomerator_df.resample('ME').sum(min_count=1)
    motnhly_semib_df = -sum_numerator_df / sum_denomerator_df.values  # sign change
    return motnhly_semib_df


def semibeta_up(r_all, signed_market_returns):
    numerator_df = r_all * signed_market_returns.values
    denomerator_df = signed_market_returns ** 2
    sum_numerator_df = numerator_df.resample('ME').sum(min_count=1)
    sum_denomerator_df = denomerator_df.resample('ME').sum(min_count=1)
    beta_up = sum_numerator_df / sum_denomerator_df.values
    return beta_up


def semibeta_down(r_all, signed_market_returns):
    numerator_df = r_all * signed_market_returns.values
    denomerator_df = signed_market_returns ** 2
    sum_numerator_df = numerator_df.resample('ME').sum(min_count=1)
    sum_denomerator_df = denomerator_df.resample('ME').sum(min_count=1)
    beta_up = sum_numerator_df / sum_denomerator_df.values
    return beta_up

# Function to winsorize column


def fn_win(Win_series, lower_limit, upper_limit):
```

```
    return pd.Series(winsorize(Win_series, limits=(lower_limit,
       upper_limit)), index=Win_series.index)


# function to calculate summary statistics
def summarystatistics_fn(motnhly_semib_df):
    crossectional_mean = motnhly_semib_df.mean(axis=1)
    crossectional_std = motnhly_semib_df.std(axis=1, ddof=1)
    crossectional_median = motnhly_semib_df.median(axis=1)
    timeseries_mean = crossectional_mean.mean()
    ts_std = crossectional_std.mean()
    ts_median = crossectional_median.mean()
    return timeseries_mean, ts_median, ts_std


def averagecross_correlation(matrix1, matrix2):
    correlations = []
    for i in range(len(matrix1)):
        corr = matrix1.iloc[i].corr(matrix2.iloc[i], 'pearson')
        correlations.append(corr)
    return np.mean(correlations)

# neweywest functions for SE and tstat


def fn_newey_se(coeff, nlags):
    res_1 = sm.OLS(coeff, np.ones(len(coeff))).fit()
    neweycov = cov_hac(res_1, nlags=nlags)
    neweyse = np.sqrt(np.diag(neweycov))
    return neweyse


def fn_newey_tstat(coeff, mean_coeff, nlags):
    neweyse = fn_newey_se(coeff, nlags)
    nw_tstat = mean_coeff / neweyse
    p_values = 2 * (1 - norm.cdf(np.abs(nw_tstat)))
    return nw_tstat, p_values


def calculate_coskewness_cokurtosis(r, m):
    r_monthly_mean = r.resample('M').mean()
    m_monthly_mean = m.resample('M').mean()
    r_diff = r.groupby(r.index.to_period(
        'M')).transform(lambda x: x - x.mean())
    m_diff = m.groupby(m.index.to_period(
        'M')).transform(lambda x: x - x.mean())
```

```
    coskewness_num = r_diff.mul(m_diff.values**2, axis=0).resample('M')
        .sum()
    coskewness_denom_r = r_diff.pow(2).resample(
        'M').sum()  # r_diff^2 sum per month
    coskewness_denom_m = m_diff.pow(2).resample(
        'M').sum()  # m_diff^2 sum per month
    coskewness = coskewness_num / \
        (coskewness_denom_r * coskewness_denom_m.values +
         1e-8)  # Avoid division by zero
    cokurtosis_num = r_diff.mul(m_diff.values**3, axis=0).resample('M')
        .sum()
    cokurtosis_denom_r = r_diff.pow(2).resample('M').sum()
    cokurtosis_denom_m = m_diff.pow(2).resample('M').sum()
    cokurtosis_denom = (cokurtosis_denom_r *
                        cokurtosis_denom_m.values).pow(3/2)
    cokurtosis = cokurtosis_num / \
        (cokurtosis_denom + 1e-8)  # Avoid division by zero
    return coskewness, cokurtosis


#
    ================================================================================
#  Cleaning data
#
    ================================================================================
file_path = r"C:\\Users\\mayan\\OneDrive - University of Birmingham\\
    Desktop\\dessertation\\data\\CRPSsmall.csv"
thedata = pd.read_csv(file_path)
thedata['date'] = pd.to_datetime(thedata['date'])

# Set 'date' as the index without removing the column
thedata.set_index('date', drop=False, inplace=True)
relevant_columns = ['PRC', 'VOL', 'SHROUT', 'DLSTCD']
thedata['PRC'] = thedata['PRC'].abs()  # Take absolute values of PRC
thedata[relevant_columns] = thedata[relevant_columns].apply(pd.
    to_numeric)

# Identify PERMNOs where PRC is less than 5 and drop them
permno_to_drop = thedata[thedata['PRC'] < 5]['PERMNO'].unique()
thedata = thedata[~thedata['PERMNO'].isin(permno_to_drop)]
thedata['DLSTCD'] = pd.to_numeric(thedata['DLSTCD'])
thedata = thedata[thedata['DLSTCD'].isna()]

# dropping shares with low trading volume
average_volume = thedata.groupby('PERMNO')['VOL'].mean()
```

```
lower_15_percentile = average_volume.quantile(0.15)
low_volume_share_1s = average_volume[average_volume <
                                     lower_15_percentile].index
thedata = thedata[~thedata['PERMNO'].isin(low_volume_share_1s)]
number_of_low_volume_share_1s = len(low_volume_share_1s)
unique_dates = thedata['date'].unique()

# Identify PERMNOs that either:
# 1. Don't have entries for all unique dates
# 2. Are missing PRC values for any date
# and dropping them
permno_to_drop = thedata.groupby('PERMNO').filter(
    lambda x: len(x['date'].unique()) < len(
        unique_dates) or x['PRC'].isna().any()
)['PERMNO'].unique()
number_of_permno_dropped = len(permno_to_drop)
print(f"Number of PERMNOs to be dropped: {number_of_permno_dropped}")
thedata = thedata[~thedata['PERMNO'].isin(permno_to_drop)]
remaining_permno_count = thedata['PERMNO'].nunique()

thedata.drop(columns=['date'], inplace=True)
thedata.reset_index(inplace=True)

# calculating log returns
thedata = thedata.sort_values(by=['PERMNO', 'date'])
thedata['RET'] = thedata.groupby('PERMNO')['PRC'].apply(
    lambda x: np.log(x / x.shift(1))
).reset_index(level=0, drop=True)

# Step 3: Remove rows where the return ('RET') is NaN (the first date
    for each share_1)
thedata = thedata[~thedata['RET'].isna()]

# Backfill 0 returns in the RET column
thedata['RET'] = thedata['RET'].replace(0, np.nan).bfill()

# Remove specified columns
columns_to_remove = ['DLSTCD', 'VOL']
thedata = thedata.drop(columns=columns_to_remove)

# preparing famafrench data
famafrenchdata = pd.read_csv(
    r"C:\\Users\\mayan\\OneDrive - University of Birmingham\\Desktop\\
        dessertation\\data\\fama fench data 1.csv")
famafrenchdata['date'] = pd.to_datetime(famafrenchdata['date'])
ff_factors = famafrenchdata[['date', 'smb', 'hml', 'umd']]
thedata['date'] = pd.to_datetime(thedata['date'])
```

```
# Merge Fama-French factors into 'thedata' on 'date'
thedata = pd.merge(thedata, ff_factors, on='date', how='left')


columns_to_drop = ['PERMCO', 'CUSIP', 'RCRDDT', 'SHRCD', 'vwretd', '
    ewretd']
thedata = thedata.drop(columns=columns_to_drop)

# calculating daily market returns
thedata['market_cap'] = thedata['PRC'] * thedata['SHROUT']
daily_market_return = thedata.groupby('date').apply(
    lambda x: np.sum(x['RET'] * x['market_cap']) / np.sum(x['market_cap
        ']))
thedata['market_return'] = thedata['date'].map(daily_market_return)


# Preparing stocks and market returns
thedata['date'] = pd.to_datetime(thedata['date'])
temp_df = thedata[['PERMNO', 'date', 'RET']]
r = temp_df.pivot_table(index='date', columns='PERMNO', values='RET')
m = thedata[['date', 'market_return']]
m = m.drop_duplicates(subset=['date'])
m.set_index('date', inplace=True)
m.columns = ['daily market ret']


r_monthly = r.resample('M').sum()


#
    ===========================================================================
# calculating factors
#
    ===========================================================================

# segregating returns into positive and negative
r_pos = r.mask(r < 0, 0)
r_neg = r.mask(r > 0, 0)
m_pos = m.mask(m < 0, 0)
m_neg = m.mask(m > 0, 0)

# Handle outlier months
r = fn_monthoutlier(r)
m = fn_monthoutlier(m)
r_pos = fn_monthoutlier(r_pos)
```

```
r_neg = fn_monthoutlier(r_neg)
m_pos = fn_monthoutlier(m_pos)
m_neg = fn_monthoutlier(m_neg)


# semibeta calculation
beta_CAPM = fn_semibeta_cal(r, m, m)
beta_N = fn_semibeta_cal(r_neg, m_neg, m)
beta_P = fn_semibeta_cal(r_pos, m_pos, m)
beta_M_pos = fn_mixed_semib(r_neg, m_pos, m)
beta_M_neg = fn_mixed_semib(r_pos, m_neg, m)

# Preparing Fama-French factors
ff_factors_monthly = thedata[['date', 'smb', 'hml', 'umd']]
ff_factors_monthly.set_index('date', inplace=True)
ff_factors_monthly = ff_factors_monthly.resample('M').mean()
ff_factors_monthly.index = ff_factors_monthly.index.to_period('M')

# Step 1: Calculate Upside and Downside Betas, Coskewness, and Cokurtosis
beta_up = semibeta_up(r, m_pos)
beta_down = semibeta_down(r, m_neg)
coskewness, cokurtosis = calculate_coskewness_cokurtosis(r, m)

# Winsorize at 1% and 99% levels
elements_to_winsorize = [beta_up, beta_down, coskewness,
                         cokurtosis, beta_CAPM, beta_N, beta_P, beta_M_pos, beta_M_neg]
for element in elements_to_winsorize:
    element.apply(fn_win, lower_limit=0.01,
                  upper_limit=0.01, axis=0)

# Convert all column names to strings and indices to PeriodIndex
dataframes_to_convert = [beta_up, beta_down, r_monthly, beta_CAPM, beta_N,
                         beta_P, beta_M_pos, beta_M_neg, coskewness, cokurtosis, ff_factors_monthly]
for df in dataframes_to_convert:
    df.columns = df.columns.astype(str)
    df.index = pd.PeriodIndex(df.index, freq='M')


# ==============================================================================
# semibeta summary stats
```

```
# ================================================================================

# Semibeta summary statistics
sumstat_index = ['Mean', 'Med', 'Std']
sumstat_col = ['B', 'B_N', 'B_P', 'B_M+', 'B_M-']
sumstat = pd.DataFrame(index=sumstat_index, columns=sumstat_col)
sumstat['B'] = summarystatistics_fn(beta_CAPM)
sumstat['B_N'] = summarystatistics_fn(beta_N)
sumstat['B_P'] = summarystatistics_fn(beta_P)
sumstat['B_M+'] = summarystatistics_fn(beta_M_pos)
sumstat['B_M-'] = summarystatistics_fn(beta_M_neg)

# Correlation between semibetas
betas = [beta_CAPM, beta_N, beta_P, beta_M_pos, beta_M_neg]
corr_matrix_label = ['B', 'B_N', 'B_P', 'B_M+', 'B_M-']
corr_matrix = pd.DataFrame(index=corr_matrix_label, columns=
    corr_matrix_label)
for i in range(len(betas)):
    for j in range(i, len(betas)):
        if i == j:
            corr_matrix.iloc[i, j] = 1.0
        else:
            flat1 = betas[i].values.flatten()
            flat2 = betas[j].values.flatten()
            combined_df = pd.DataFrame({'flat1': flat1, 'flat2': flat2
                })
            clean_df = combined_df.dropna()
            corr = np.corrcoef(clean_df['flat1'], clean_df['flat2'])[0,
                 1]
            corr_matrix.iloc[j, i] = corr


print("Semibeta Summary Statistics")
print(sumstat.to_string())
print("\nSemibeta Correlation Matrix")
print(corr_matrix.to_string())


# ================================================================================
# regression 1 - CAPM
# ================================================================================
```

```python

lambda_0_reg1, lambda_CAPM_reg1, rsquared_reg1 = [], [], []
models = []
regression_counts = {}

for share_1 in beta_CAPM.columns:
    beta_values = beta_CAPM[share_1]
    r_values = r_monthly[share_1]

    # Align the indices
    aligned_data = pd.concat([beta_values, r_values],
                             axis=1, keys=['CAPM', 'r']).dropna()

    # If there's not enough data after alignment, skip this share

    if aligned_data.empty or len(aligned_data) < 2:
        print(f"Skipping share_1 {share_1} due to insufficient data.")
        continue

    x = sm.add_constant(aligned_data[['CAPM']], has_constant='add')
    y = aligned_data['r']
    model_reg1 = sm.OLS(y, x).fit()

    # Store the regression res_1
    models.append(model_reg1)
    lambda_0_reg1.append(model_reg1.params.get('const', np.nan))
    lambda_CAPM_reg1.append(model_reg1.params.get('CAPM', np.nan))
    rsquared_reg1.append(model_reg1.rsquared)
    regression_counts[share_1] = len(aligned_data)

coeff_reg1 = pd.DataFrame(
    {'lambda_0': lambda_0_reg1, 'lambda_CAPM': lambda_CAPM_reg1})

mean_coeff_reg1 = coeff_reg1.mean()

lambda_0_coef_reg1 = mean_coeff_reg1['lambda_0'] * 100
lambda_CAPM_coef_reg1 = mean_coeff_reg1['lambda_CAPM'] * 100

# Calculate Newey-West t-statistics
nlags = round(0.75 * len(r_monthly) ** (1/3))
nw_tstat0, p_values0 = fn_newey_tstat(
    coeff_reg1['lambda_0'], mean_coeff_reg1['lambda_0'], nlags)
nw_tstat1, p_values1 = fn_newey_tstat(
    coeff_reg1['lambda_CAPM'], mean_coeff_reg1['lambda_CAPM'], nlags)

# Calculate the average R-squared value
rsquared_mean_reg1 = np.nanmean(rsquared_reg1) * 100
```

```

# Prepare the res_1 for display in a DataFrame
regression_res_1 = pd.DataFrame({
    'Variable': ['Constant', 'CAPM'],
    'Coefficient (%)': [lambda_0_coef_reg1, lambda_CAPM_coef_reg1],
    't-Statistic': [nw_tstat0[0], nw_tstat1[0]],
    'p-Value': [p_values0[0], p_values1[0]],
    'R-squared (%)': [rsquared_mean_reg1, '']  # R-squared for one row
})

# Print the res_1 in tabular form
print("Regression res_1:")
print(regression_res_1.to_string(index=False))


# ==============================================================================
# Reg 2   4semibetas
# ==============================================================================
# Initialize lists to store the regression res_1
lambda_0_list = []
lambda_N_list = []
lambda_P_list = []
lambda_M_pos_list = []
lambda_M_neg_list = []
rsquared_list = []

# Lists for storing t-statistics and p-values
tstat_lambda_0_list = []
tstat_lambda_N_list = []
tstat_lambda_P_list = []
tstat_lambda_M_pos_list = []
tstat_lambda_M_neg_list = []

pvalue_lambda_0_list = []
pvalue_lambda_N_list = []
pvalue_lambda_P_list = []
pvalue_lambda_M_pos_list = []
pvalue_lambda_M_neg_list = []

# Loop through each share in r_monthly
for share_1 in r_monthly.columns:
    N_values = beta_N[share_1]
    P_values = beta_P[share_1]
```

```
    M_pos_values = beta_M_pos[share_1]
    M_neg_values = beta_M_neg[share_1]
    r_values = r_monthly[share_1]

    # Align the data (ensure all Win_series have the same dates and
        remove NaN values)
    temp_df = pd.concat([N_values, P_values, M_pos_values, M_neg_values
        , r_values],
                        axis=1, keys=['N', 'P', 'M_pos', 'M_neg', 'r'])
                            .dropna()

    # If there's not enough data after alignment, skip this share_1
    if len(temp_df) < 2:
        continue

    X = sm.add_constant(temp_df[['N', 'P', 'M_pos', 'M_neg']])
    y = temp_df['r']  # r_{t+1}
    model = sm.OLS(y, X).fit()

    # Store the regression res_1 (coefficients and t-statistics)
    lambda_0_list.append(model.params['const'])
    lambda_N_list.append(model.params['N'])
    lambda_P_list.append(model.params['P'])
    lambda_M_pos_list.append(model.params['M_pos'])
    lambda_M_neg_list.append(model.params['M_neg'])
    rsquared_list.append(model.rsquared)

    # Store the t-statistics
    tstat_lambda_0_list.append(model.tvalues['const'])
    tstat_lambda_N_list.append(model.tvalues['N'])
    tstat_lambda_P_list.append(model.tvalues['P'])
    tstat_lambda_M_pos_list.append(model.tvalues['M_pos'])
    tstat_lambda_M_neg_list.append(model.tvalues['M_neg'])

    # Store the p-values
    pvalue_lambda_0_list.append(model.pvalues['const'])
    pvalue_lambda_N_list.append(model.pvalues['N'])
    pvalue_lambda_P_list.append(model.pvalues['P'])
    pvalue_lambda_M_pos_list.append(model.pvalues['M_pos'])
    pvalue_lambda_M_neg_list.append(model.pvalues['M_neg'])

# Convert lists to DataFrames for easy analysis
res_1_df = pd.DataFrame({
    'lambda_0': lambda_0_list,
    'lambda_N': lambda_N_list,
    'lambda_P': lambda_P_list,
    'lambda_M_pos': lambda_M_pos_list,
```

```python
    'lambda_M_neg': lambda_M_neg_list,
    'rsquared': rsquared_list
})

tstats_df = pd.DataFrame({
    'tstat_lambda_0': tstat_lambda_0_list,
    'tstat_lambda_N': tstat_lambda_N_list,
    'tstat_lambda_P': tstat_lambda_P_list,
    'tstat_lambda_M_pos': tstat_lambda_M_pos_list,
    'tstat_lambda_M_neg': tstat_lambda_M_neg_list
})

pvalues_df = pd.DataFrame({
    'pvalue_lambda_0': pvalue_lambda_0_list,
    'pvalue_lambda_N': pvalue_lambda_N_list,
    'pvalue_lambda_P': pvalue_lambda_P_list,
    'pvalue_lambda_M_pos': pvalue_lambda_M_pos_list,
    'pvalue_lambda_M_neg': pvalue_lambda_M_neg_list
})

# Calculate the average of the regression coefficients, t-statistics,
    and p-values
mean_res_1 = res_1_df.mean()
mean_tstats = tstats_df.mean()
mean_pvalues = pvalues_df.mean()

print("Average Regression res_1:")
print(mean_res_1)
print("\nAverage t-Statistics:")
print(mean_tstats)
print("\nAverage p-Values:")
print(mean_pvalues)


#
    ===========================================================================

# Reg 4 fama factors with semibetas
#
    ===========================================================================


# Initialize lists to store regression res_1
lambda_0_reg3 = []
lambda_N_reg3 = []
lambda_P_reg3 = []
lambda_M_pos_reg3 = []
```

```python
lambda_M_neg_reg3 = []
lambda_smb_reg3 = []
lambda_hml_reg3 = []
lambda_umd_reg3 = []
rsquared_reg3 = []

# Track the number of regressions run
regression_counts_reg3 = {}

# Loop through each share_1 (PERMNO) in r_monthly
for share_1 in r_monthly.columns:
    N_values = beta_N[share_1]
    P_values = beta_P[share_1]
    M_pos_values = beta_M_pos[share_1]
    M_neg_values = beta_M_neg[share_1]
    r_values = r_monthly[share_1]

    temp_df = pd.concat([N_values, P_values, M_pos_values, M_neg_values, r_values,
                         ff_factors_monthly['smb'], ff_factors_monthly['hml'], ff_factors_monthly['umd']],
                        axis=1, keys=['N', 'P', 'M_pos', 'M_neg', 'r', 'smb', 'hml', 'umd']).dropna()

    if temp_df.empty or len(temp_df) < 2:
        print(f"Skipping share_1 {share_1} due to insufficient data.")
        continue

    scaler = StandardScaler()
    independent_vars = temp_df[[
        'N', 'P', 'M_pos', 'M_neg', 'smb', 'hml', 'umd']]
    independent_vars_scaled = pd.DataFrame(scaler.fit_transform(independent_vars),
                                           columns=independent_vars.columns, index=independent_vars.index)

    temp_df['r'] = scaler.fit_transform(temp_df[['r']])

    x = sm.add_constant(independent_vars_scaled)
    y = temp_df['r']

    model_reg3 = sm.OLS(y, x).fit()

    lambda_0_reg3.append(model_reg3.params.get('const', np.nan))
    lambda_N_reg3.append(model_reg3.params.get('N', np.nan))
    lambda_P_reg3.append(model_reg3.params.get('P', np.nan))
```

```
        lambda_M_pos_reg3.append(model_reg3.params.get('M_pos', np.nan))
        lambda_M_neg_reg3.append(model_reg3.params.get('M_neg', np.nan))
        lambda_smb_reg3.append(model_reg3.params.get('smb', np.nan))
        lambda_hml_reg3.append(model_reg3.params.get('hml', np.nan))
        lambda_umd_reg3.append(model_reg3.params.get('umd', np.nan))
        rsquared_reg3.append(model_reg3.rsquared)

        regression_counts_reg3[share_1] = len(temp_df)

coeff_reg3 = pd.DataFrame({
    'lambda_0': lambda_0_reg3,
    'lambda_N': lambda_N_reg3,
    'lambda_P': lambda_P_reg3,
    'lambda_M_pos': lambda_M_pos_reg3,
    'lambda_M_neg': lambda_M_neg_reg3,
    'lambda_smb': lambda_smb_reg3,
    'lambda_hml': lambda_hml_reg3,
    'lambda_umd': lambda_umd_reg3
})

mean_coeff_reg3 = coeff_reg3.mean()

lambda_0_coef_reg3 = mean_coeff_reg3['lambda_0'] * 100
lambda_N_coef_reg3 = mean_coeff_reg3['lambda_N'] * 100
lambda_P_coef_reg3 = mean_coeff_reg3['lambda_P'] * 100
lambda_M_pos_coef_reg3 = mean_coeff_reg3['lambda_M_pos'] * 100
lambda_M_neg_coef_reg3 = mean_coeff_reg3['lambda_M_neg'] * 100
lambda_smb_coef_reg3 = mean_coeff_reg3['lambda_smb'] * 100
lambda_hml_coef_reg3 = mean_coeff_reg3['lambda_hml'] * 100
lambda_umd_coef_reg3 = mean_coeff_reg3['lambda_umd'] * 100

nlags = round(0.75 * len(r_monthly) ** (1/3))
nw_tstat0_reg3, p_values0_reg3 = fn_newey_tstat(
    coeff_reg3['lambda_0']/10, mean_coeff_reg3['lambda_0'], nlags)
nw_tstat_N_reg3, p_values_N_reg3 = fn_newey_tstat(
    coeff_reg3['lambda_N'], mean_coeff_reg3['lambda_N'], nlags)
nw_tstat_P_reg3, p_values_P_reg3 = fn_newey_tstat(
    coeff_reg3['lambda_P'], mean_coeff_reg3['lambda_P'], nlags)
nw_tstat_M_pos_reg3, p_values_M_pos_reg3 = fn_newey_tstat(
    coeff_reg3['lambda_M_pos'], mean_coeff_reg3['lambda_M_pos'], nlags)
nw_tstat_M_neg_reg3, p_values_M_neg_reg3 = fn_newey_tstat(
    coeff_reg3['lambda_M_neg'], mean_coeff_reg3['lambda_M_neg'], nlags)
nw_tstat_smb_reg3, p_values_smb_reg3 = fn_newey_tstat(
    coeff_reg3['lambda_smb'], mean_coeff_reg3['lambda_smb'], nlags)
nw_tstat_hml_reg3, p_values_hml_reg3 = fn_newey_tstat(
    coeff_reg3['lambda_hml'], mean_coeff_reg3['lambda_hml'], nlags)
nw_tstat_umd_reg3, p_values_umd_reg3 = fn_newey_tstat(
```

```
    coeff_reg3['lambda_umd'], mean_coeff_reg3['lambda_umd'], nlags)


rsquared_mean_reg3 = np.mean(rsquared_reg3) * 100

regression_res_1_reg3 = pd.DataFrame({
    'Variable': ['Constant', 'N', 'P', 'M_pos', 'M_neg', 'smb', 'hml', 'umd'],
    'Coefficient (%)': [lambda_0_coef_reg3, lambda_N_coef_reg3, lambda_P_coef_reg3,
                        lambda_M_pos_coef_reg3, lambda_M_neg_coef_reg3,
                        lambda_smb_coef_reg3, lambda_hml_coef_reg3, lambda_umd_coef_reg3],
    't-Statistic': [nw_tstat0_reg3[0], nw_tstat_N_reg3[0], nw_tstat_P_reg3[0],
                    nw_tstat_M_pos_reg3[0], nw_tstat_M_neg_reg3[0],
                    nw_tstat_smb_reg3[0], nw_tstat_hml_reg3[0], nw_tstat_umd_reg3[0]],
    'p-Value': [p_values0_reg3[0], p_values_N_reg3[0], p_values_P_reg3[0],
                p_values_M_pos_reg3[0], p_values_M_neg_reg3[0],
                p_values_smb_reg3[0], p_values_hml_reg3[0], p_values_umd_reg3[0]],
    # R-squared only in the first row
    'R-squared (%)': [rsquared_mean_reg3, '', '', '', '', '', '', '']
})

print("Regression res_1 with Standardized Fama-French and Semibetas:")
print(regression_res_1_reg3.to_string(index=False))


# =============================================================================
# reg 4 beta up and down
# =============================================================================

lambda_0_list = []
lambda_up_list = []
lambda_down_list = []
rsquared_list = []

tstat_lambda_0_list = []
tstat_lambda_up_list = []
tstat_lambda_down_list = []
```

```python

pvalue_lambda_0_list = []
pvalue_lambda_up_list = []
pvalue_lambda_down_list = []

for share_1 in r_monthly.columns:
    up_values = beta_up[share_1]
    down_values = beta_down[share_1]
    r_values = r_monthly[share_1]

    temp_df = pd.concat([up_values, down_values, r_values],
                        axis=1, keys=['up', 'down', 'r']).dropna()

    if len(temp_df) < 2:
        continue

    X = sm.add_constant(temp_df[['up', 'down']])
    y = temp_df['r']  # r_{t+1}

    model = sm.OLS(y, X).fit()

    lambda_0_list.append(model.params['const'])
    lambda_up_list.append(model.params['up'])
    lambda_down_list.append(model.params['down'])
    rsquared_list.append(model.rsquared)

    tstat_lambda_0_list.append(model.tvalues['const'])
    tstat_lambda_up_list.append(model.tvalues['up'])
    tstat_lambda_down_list.append(model.tvalues['down'])

    pvalue_lambda_0_list.append(model.pvalues['const'])
    pvalue_lambda_up_list.append(model.pvalues['up'])
    pvalue_lambda_down_list.append(model.pvalues['down'])

res_1_df = pd.DataFrame({
    'lambda_0': lambda_0_list,
    'lambda_up': lambda_up_list,
    'lambda_down': lambda_down_list,
    'rsquared': rsquared_list
})

tstats_df = pd.DataFrame({
    'tstat_lambda_0': tstat_lambda_0_list,
    'tstat_lambda_up': tstat_lambda_up_list,
    'tstat_lambda_down': tstat_lambda_down_list
})

```

```python
pvalues_df = pd.DataFrame({
    'pvalue_lambda_0': pvalue_lambda_0_list,
    'pvalue_lambda_up': pvalue_lambda_up_list,
    'pvalue_lambda_down': pvalue_lambda_down_list
})

mean_res_1 = res_1_df.mean()
mean_tstats = tstats_df.mean()
mean_pvalues = pvalues_df.mean()

print("Average Regression res_1:")
print(mean_res_1)
print("\nAverage t-Statistics:")
print(mean_tstats)
print("\nAverage p-Values:")
print(mean_pvalues)


# ==============================================================================
# reg 5 beta up down with semibeta
# ==============================================================================
lambda_0_list = []
lambda_up_list = []
lambda_down_list = []
lambda_N_list = []
lambda_P_list = []
lambda_M_pos_list = []
lambda_M_neg_list = []
rsquared_list = []

tstat_lambda_0_list = []
tstat_lambda_up_list = []
tstat_lambda_down_list = []
tstat_lambda_N_list = []
tstat_lambda_P_list = []
tstat_lambda_M_pos_list = []
tstat_lambda_M_neg_list = []

pvalue_lambda_0_list = []
pvalue_lambda_up_list = []
pvalue_lambda_down_list = []
pvalue_lambda_N_list = []
pvalue_lambda_P_list = []
```

```python
pvalue_lambda_M_pos_list = []
pvalue_lambda_M_neg_list = []

# Loop through each share_1 (PERMNO) in r_monthly
for share_1 in r_monthly.columns:
    # Extract the data for the current share_1 from all DataFrames
    up_values = beta_up[share_1]
    down_values = beta_down[share_1]
    N_values = beta_N[share_1]
    P_values = beta_P[share_1]
    M_pos_values = beta_M_pos[share_1]
    M_neg_values = beta_M_neg[share_1]
    r_values = r_monthly[share_1]

    # Align the data (ensure all Win_series have the same dates and remove NaN values)
    temp_df = pd.concat([up_values, down_values, N_values, P_values, M_pos_values, M_neg_values, r_values],
                        axis=1, keys=['up', 'down', 'N', 'P', 'M_pos', 'M_neg', 'r']).dropna()

    # If there's not enough data after alignment, skip this share
    if len(temp_df) < 2:
        continue

    X = sm.add_constant(temp_df[['up', 'down', 'N', 'P', 'M_pos', 'M_neg']])
    y = temp_df['r']
    model = sm.OLS(y, X).fit()

    lambda_0_list.append(model.params['const'])
    lambda_up_list.append(model.params['up'])
    lambda_down_list.append(model.params['down'])
    lambda_N_list.append(model.params['N'])
    lambda_P_list.append(model.params['P'])
    lambda_M_pos_list.append(model.params['M_pos'])
    lambda_M_neg_list.append(model.params['M_neg'])
    rsquared_list.append(model.rsquared)

    tstat_lambda_0_list.append(model.tvalues['const'])
    tstat_lambda_up_list.append(model.tvalues['up'])
    tstat_lambda_down_list.append(model.tvalues['down'])
    tstat_lambda_N_list.append(model.tvalues['N'])
    tstat_lambda_P_list.append(model.tvalues['P'])
    tstat_lambda_M_pos_list.append(model.tvalues['M_pos'])
    tstat_lambda_M_neg_list.append(model.tvalues['M_neg'])

```

```
    pvalue_lambda_0_list.append(model.pvalues['const'])
    pvalue_lambda_up_list.append(model.pvalues['up'])
    pvalue_lambda_down_list.append(model.pvalues['down'])
    pvalue_lambda_N_list.append(model.pvalues['N'])
    pvalue_lambda_P_list.append(model.pvalues['P'])
    pvalue_lambda_M_pos_list.append(model.pvalues['M_pos'])
    pvalue_lambda_M_neg_list.append(model.pvalues['M_neg'])

res_1_df = pd.DataFrame({
    'lambda_0': lambda_0_list,
    'lambda_up': lambda_up_list,
    'lambda_down': lambda_down_list,
    'lambda_N': lambda_N_list,
    'lambda_P': lambda_P_list,
    'lambda_M_pos': lambda_M_pos_list,
    'lambda_M_neg': lambda_M_neg_list,
    'rsquared': rsquared_list
})

tstats_df = pd.DataFrame({
    'tstat_lambda_0': tstat_lambda_0_list,
    'tstat_lambda_up': tstat_lambda_up_list,
    'tstat_lambda_down': tstat_lambda_down_list,
    'tstat_lambda_N': tstat_lambda_N_list,
    'tstat_lambda_P': tstat_lambda_P_list,
    'tstat_lambda_M_pos': tstat_lambda_M_pos_list,
    'tstat_lambda_M_neg': tstat_lambda_M_neg_list
})

pvalues_df = pd.DataFrame({
    'pvalue_lambda_0': pvalue_lambda_0_list,
    'pvalue_lambda_up': pvalue_lambda_up_list,
    'pvalue_lambda_down': pvalue_lambda_down_list,
    'pvalue_lambda_N': pvalue_lambda_N_list,
    'pvalue_lambda_P': pvalue_lambda_P_list,
    'pvalue_lambda_M_pos': pvalue_lambda_M_pos_list,
    'pvalue_lambda_M_neg': pvalue_lambda_M_neg_list
})

mean_res_1 = res_1_df.mean()
mean_tstats = tstats_df.mean()
mean_pvalues = pvalues_df.mean()

# Print the average regression res_1, t-statistics, and p-values
print("Average Regression res_1 for Regression 5:")
print(mean_res_1)
print("\nAverage t-Statistics for Regression 5:")
```

```
print(mean_tstats)
print("\nAverage p-Values for Regression 5:")
print(mean_pvalues)


# =============================================================================
# Reg6 coskew cokurt
# =============================================================================
lambda_0_list = []
lambda_coskewness_list = []
lambda_cokurtosis_list = []
rsquared_list = []

tstat_lambda_0_list = []
tstat_lambda_coskewness_list = []
tstat_lambda_cokurtosis_list = []

pvalue_lambda_0_list = []
pvalue_lambda_coskewness_list = []
pvalue_lambda_cokurtosis_list = []

for share_1 in r_monthly.columns:
    # Extract the data for the current share_1 from all DataFrames
    if share_1 not in coskewness.columns or share_1 not in cokurtosis.columns:
        continue

    coskewness_values = coskewness[share_1]
    cokurtosis_values = cokurtosis[share_1]
    r_values = r_monthly[share_1]

    temp_df = pd.concat([coskewness_values, cokurtosis_values, r_values],
                        axis=1, keys=['coskewness', 'cokurtosis', 'r']).dropna()

    if len(temp_df) < 2:
        continue

    X = sm.add_constant(temp_df[['coskewness', 'cokurtosis']])
    y = temp_df['r']  # r_{t+1}

    model = sm.OLS(y, X).fit()
```

```
    lambda_0_list.append(model.params['const'])
    lambda_coskewness_list.append(model.params['coskewness'])
    lambda_cokurtosis_list.append(model.params['cokurtosis'])
    rsquared_list.append(model.rsquared)
    tstat_lambda_0_list.append(model.tvalues['const'])
    tstat_lambda_coskewness_list.append(model.tvalues['coskewness'])
    tstat_lambda_cokurtosis_list.append(model.tvalues['cokurtosis'])
    pvalue_lambda_0_list.append(model.pvalues['const'])
    pvalue_lambda_coskewness_list.append(model.pvalues['coskewness'])
    pvalue_lambda_cokurtosis_list.append(model.pvalues['cokurtosis'])

res_1_df = pd.DataFrame({
    'lambda_0': lambda_0_list,
    'lambda_coskewness': lambda_coskewness_list,
    'lambda_cokurtosis': lambda_cokurtosis_list,
    'rsquared': rsquared_list
})

tstats_df = pd.DataFrame({
    'tstat_lambda_0': tstat_lambda_0_list,
    'tstat_lambda_coskewness': tstat_lambda_coskewness_list,
    'tstat_lambda_cokurtosis': tstat_lambda_cokurtosis_list
})

pvalues_df = pd.DataFrame({
    'pvalue_lambda_0': pvalue_lambda_0_list,
    'pvalue_lambda_coskewness': pvalue_lambda_coskewness_list,
    'pvalue_lambda_cokurtosis': pvalue_lambda_cokurtosis_list
})

mean_res_1 = res_1_df.mean()
mean_tstats = tstats_df.mean()
mean_pvalues = pvalues_df.mean()

print("Average Regression res_1 for Regression 6:")
print(mean_res_1)
print("\nAverage t-Statistics for Regression 6:")
print(mean_tstats)
print("\nAverage p-Values for Regression 6:")
print(mean_pvalues)


#
    ==============================================================================
# reg7 coskew cokurt with semibeta
```

```python
# ==========================================================================================

lambda_0_list = []
lambda_coskewness_list = []
lambda_cokurtosis_list = []
lambda_N_list = []
lambda_P_list = []
lambda_M_pos_list = []
lambda_M_neg_list = []
rsquared_list = []

tstat_lambda_0_list = []
tstat_lambda_coskewness_list = []
tstat_lambda_cokurtosis_list = []
tstat_lambda_N_list = []
tstat_lambda_P_list = []
tstat_lambda_M_pos_list = []
tstat_lambda_M_neg_list = []

pvalue_lambda_0_list = []
pvalue_lambda_coskewness_list = []
pvalue_lambda_cokurtosis_list = []
pvalue_lambda_N_list = []
pvalue_lambda_P_list = []
pvalue_lambda_M_pos_list = []
pvalue_lambda_M_neg_list = []

# Loop through each share_1 (PERMNO) in r_monthly
for share_1 in r_monthly.columns:
    # Extract the data for the current share_1 from all DataFrames
    if share_1 not in coskewness.columns or share_1 not in cokurtosis.columns or \
       share_1 not in beta_N.columns or share_1 not in beta_P.columns or \
       share_1 not in beta_M_pos.columns or share_1 not in beta_M_neg.columns:
        continue

    coskewness_values = coskewness[share_1]
    cokurtosis_values = cokurtosis[share_1]
    N_values = beta_N[share_1]
    P_values = beta_P[share_1]
    M_pos_values = beta_M_pos[share_1]
    M_neg_values = beta_M_neg[share_1]
    r_values = r_monthly[share_1]
```

```python

    # Align the data (ensure all Win_series have the same dates and remove NaN values)
    temp_df = pd.concat([coskewness_values, cokurtosis_values, N_values, P_values, M_pos_values, M_neg_values, r_values],
                        axis=1, keys=['coskewness', 'cokurtosis', 'N', 'P', 'M_pos', 'M_neg', 'r']).dropna()

    # If there's not enough data after alignment, skip this share_1
    if len(temp_df) < 2:
        continue

    X = sm.add_constant(
        temp_df[['coskewness', 'cokurtosis', 'N', 'P', 'M_pos', 'M_neg']])
    y = temp_df['r']  # r_{t+1}

    model = sm.OLS(y, X).fit()
    lambda_0_list.append(model.params['const'])
    lambda_coskewness_list.append(model.params['coskewness'])
    lambda_cokurtosis_list.append(model.params['cokurtosis'])
    lambda_N_list.append(model.params['N'])
    lambda_P_list.append(model.params['P'])
    lambda_M_pos_list.append(model.params['M_pos'])
    lambda_M_neg_list.append(model.params['M_neg'])
    rsquared_list.append(model.rsquared)
    tstat_lambda_0_list.append(model.tvalues['const'])
    tstat_lambda_coskewness_list.append(model.tvalues['coskewness'])
    tstat_lambda_cokurtosis_list.append(model.tvalues['cokurtosis'])
    tstat_lambda_N_list.append(model.tvalues['N'])
    tstat_lambda_P_list.append(model.tvalues['P'])
    tstat_lambda_M_pos_list.append(model.tvalues['M_pos'])
    tstat_lambda_M_neg_list.append(model.tvalues['M_neg'])
    pvalue_lambda_0_list.append(model.pvalues['const'])
    pvalue_lambda_coskewness_list.append(model.pvalues['coskewness'])
    pvalue_lambda_cokurtosis_list.append(model.pvalues['cokurtosis'])
    pvalue_lambda_N_list.append(model.pvalues['N'])
    pvalue_lambda_P_list.append(model.pvalues['P'])
    pvalue_lambda_M_pos_list.append(model.pvalues['M_pos'])
    pvalue_lambda_M_neg_list.append(model.pvalues['M_neg'])

# Convert lists to DataFrames for easy analysis
res_1_df = pd.DataFrame({
    'lambda_0': lambda_0_list,
    'lambda_coskewness': lambda_coskewness_list,
    'lambda_cokurtosis': lambda_cokurtosis_list,
    'lambda_N': lambda_N_list,
```

```
    'lambda_P': lambda_P_list,
    'lambda_M_pos': lambda_M_pos_list,
    'lambda_M_neg': lambda_M_neg_list,
    'rsquared': rsquared_list
})

tstats_df = pd.DataFrame({
    'tstat_lambda_0': tstat_lambda_0_list,
    'tstat_lambda_coskewness': tstat_lambda_coskewness_list,
    'tstat_lambda_cokurtosis': tstat_lambda_cokurtosis_list,
    'tstat_lambda_N': tstat_lambda_N_list,
    'tstat_lambda_P': tstat_lambda_P_list,
    'tstat_lambda_M_pos': tstat_lambda_M_pos_list,
    'tstat_lambda_M_neg': tstat_lambda_M_neg_list
})

pvalues_df = pd.DataFrame({
    'pvalue_lambda_0': pvalue_lambda_0_list,
    'pvalue_lambda_coskewness': pvalue_lambda_coskewness_list,
    'pvalue_lambda_cokurtosis': pvalue_lambda_cokurtosis_list,
    'pvalue_lambda_N': pvalue_lambda_N_list,
    'pvalue_lambda_P': pvalue_lambda_P_list,
    'pvalue_lambda_M_pos': pvalue_lambda_M_pos_list,
    'pvalue_lambda_M_neg': pvalue_lambda_M_neg_list
})

mean_res_1 = res_1_df.mean()
mean_tstats = tstats_df.mean()
mean_pvalues = pvalues_df.mean()

print("Average Regression res_1 for Regression with Coskewness, Cokurtosis, and Semibetas:")
print(mean_res_1)
print("\nAverage t-Statistics for Regression:")
print(mean_tstats)
print("\nAverage p-Values for Regression:")
print(mean_pvalues)


# ==============================================================================
# Figures- density of semibetas and correlation heatmap
# ==============================================================================

```

```python
# density plot
sns.set(style="whitegrid")
plt.figure(figsize=(10, 6))
sns.kdeplot(beta_CAPM.values.flatten(),
            label=r'$\beta$', color='#FF6347', lw=2)
sns.kdeplot(beta_N.values.flatten(), label=r'$\beta^N$',
            linestyle='--', lw=2, color='#1E90FF')
sns.kdeplot(beta_P.values.flatten(), label=r'$\beta^P$',
            linestyle='-.', lw=2, color='#FFA500')
sns.kdeplot(beta_M_pos.values.flatten(),
            label=r'$\beta^{M+}$', linestyle=':', lw=2, color='#32CD32'
               )
sns.kdeplot(beta_M_neg.values.flatten(),
            label=r'$\beta^{M-}$', linestyle='-', lw=2, color='#00CED1'
               )

plt.xlim(-1, 3)  # Adjust this based on your data distribution

plt.title("Panel A: Distribution of Semibetas", fontsize=16, fontweight
   ='bold')
plt.xlabel(r'$\beta$', fontsize=14)
plt.ylabel("Density", fontsize=14)
plt.legend(loc="upper right", fontsize=12)
plt.grid(True, which='both', linestyle='--', linewidth=0.5)
sns.despine()

plt.tight_layout()
plt.show()


# corrlation heatmap
for i in range(len(betas)):
    for j in range(i + 1, len(betas)):
        # Copy upper triangle to lower triangle
        corr_matrix.iloc[i, j] = corr_matrix.iloc[j, i]

plt.figure(figsize=(10, 6))  # Set the size of the heatmap
sns.heatmap(corr_matrix.astype(float), annot=True,
            cmap='coolwarm', center=0, fmt=".2f", linewidths=0.5)

# Adding title and labels
plt.title("Correlation Heatmap of Semibetas")
plt.show()


# wald test

```

```
# Defining the null hypothesis constraints matrix
# The constraints matrix should be constructed based on the hypothesis
    H0: lambda_N = lambda_P = -lambda_M_pos = -lambda_M_neg
R = np.array([
    [1, -1, 0, 0],    # Tests lambda_N = lambda_P
    [1, 0, 1, 0],    # Tests lambda_N = -lambda_M_pos
    [1, 0, 0, 1]     # Tests lambda_N = -lambda_M_neg
])

# Mean values of the coefficients from the res_1
mean_coefficients = np.array([
    mean_res_1['lambda_N'],
    mean_res_1['lambda_P'],
    mean_res_1['lambda_M_pos'],
    mean_res_1['lambda_M_neg']
])

cov_matrix = np.cov([lambda_N_list, lambda_P_list,
                    lambda_M_pos_list, lambda_M_neg_list])

# Calculate the Wald statistic
diff = R @ mean_coefficients  # (R*Beta)
# (R*Beta)' (R*Cov*R')^-1 (R*Beta)
wald_stat = diff.T @ np.linalg.inv(R @ cov_matrix @ R.T) @ diff

# Degrees of freedom is the number of constraints (rows in matrix R)
df = R.shape[0]

# Calculate the p-value for the Wald test statistic
p_value = chi2.sf(wald_stat, df)

print("Wald Test Statistic:", wald_stat)
print("Degrees of Freedom:", df)
print("p-Value:", p_value)
```