LAPIN AMK
Lapland University of Applied Sciences

Bayes-Stein Estimators in Mean-Semivariance Portfolio Optimization

Tcysin, Aleksei
Pivovarov, Dmitrii

Thesis
Lapland University of Applied Sciences
International Business
Bachelor of Business Administration

2017

LAPIN AMK
Lapland University of Applied Sciences

Abstract of Thesis

Lapland University of Applied Sciences
International Business
Bachelor of Business Administration

Authors Aleksei Tcysin, Pivovarov Dmitrii 2017
Supervisor Antti Ovaskainen
Commissioned by
Title of Thesis Bayes-Stein Estimators in Mean-Semivariance Portfolio Optimization
Number of pages 62 + 9

The aim of this thesis was to design and implement a semivariance-based portfolio optimization model with application of shrinkage estimators.

Harry Markowitz' Modern Portfolio Theory served as a basic theoretical framework; it is further extended by using semivariance computational procedure proposed by Javier Estrada and adjusting the vector of expected returns with Bayes-Stein estimator, suggested by Philippe Jorion.

Back testing was applied in order to check the performance of the suggested scheme. Investment strategy was tested on 30 stocks representing Dow Jones Industrial Average; minimum-risk mean-variance portfolio and S&P 500 index were used as performance benchmarks. Three sets of tests were conducted to check the model in various market conditions.

Results indicate that the proposed framework is somewhat successful, outperforming both benchmarks in bullish and stagnant environments. However, further experiments under various conditions and parameters are necessary before utilizing the suggested approach in practice. Therefore, future work concerns mostly testing and evaluation routines.

Key words: semivariance, shrinkage, portfolio optimization

SYMBOLS AND ABBREVIATIONS

EMH	Efficient Market Hypothesis
MM	Money Market
MPT	Modern Portfolio Theory
MS	Mean-Semivariance
MV	Mean-Variance

CONTENTS

1 INTRODUCTION AND OVERVIEW

1.1 Introduction

Portfolio optimization concerns the choice of various financial assets to be held in such a way as to make the portfolio more efficient than any other one according to some metrics. This problem resides on the intersection of various domains such as Investment Theory, Optimization and Statistics, making it rather complex and interesting at the same time.

The aim of this thesis is to create an investment system by extending modern portfolio theory with enhanced risk and return measures. In order to test the suggested approach, back testing procedure is used.

The approach taken in this paper seems to be a prospective addition to the area of portfolio optimization problems. The suggestion is that, with the right application of financial theories and statistical tools, it is possible to effectively boost mean-variance optimization framework based on modern portfolio theory.

By employing semi-covariance computational procedures in conjunction with shrinkage, this paper aims to build a relatively simple yet powerful system that is able to outperform commonly accepted investing alternatives. It is worth noting that this work is engineer-oriented rather than rigorously scientific; however, scientific principles are still applied when prototyping and testing.

1.2 Goals

The main goal of this thesis is to design an optimization framework based on semivariance and Bayes-Stein shrinkage estimators, and then implement the proposed scheme in a programming language of interest.

It is intended to be used as an investment tool to optimally allocate capital among the stock universe of interest. As such, the system was built to be easily extendable, computationally fast and robust. As this feat was the first and therefore educational attempt at constructing investment frameworks, the prototype is inherently simplistic as it employs a single-period investment scheme and assumes a number of constraints to simplify the design aspect.

1.3 Methodology

A constructive researching method was adopted in order to reach the goals of the project.

In order to build the necessary theoretical framework, numerous sources of information and search methods are used. Scientific databases and various publications were the main sources of information. Various e-journals, on-line books, hard-cover books, scientific journals and specialized forum are consulted. Some rather exotic sources of information such as GitHub code repositories and YouTube tutorial videos are conferred as well.

With aim to test the proposed schemes historical data sources were used to collect the records needed. Yahoo Finance and Bloomberg L.P. are employed in this particular model, though this need not be the case. Historical data for US equities and US futures since 2002 is fetched via Quantopian Inc.

1.4 Thesis Structure

Chapter 2 gives a brief overview of financial and mathematical concepts needed to at least conceptually understand the rest of the thesis.

Chapter 3 presents the high-level overview of the suggested model and explains some technical details.

The results of backtesting are presented in Chapter 4.

Finally, in Chapter 5 the whole project is summarized; the results are analyzed, constraints mentioned and directions for further research are given.

2 THEORETICAL BACKGROUND

2.1 Financial and Investment Theory

2.1.1 Financial Markets

Financial market is a place which allows people to trade various securities and fungible assets: bonds, stocks, derivatives and money. Distinctive features of financial markets are regulation of trading process, low transaction costs and wide range of financial products. Financial markets provide channelling function between players of the market (those could be organizations or individuals with superfluity or deficit of available funds). Assets may be exchanged in two forms - direct finance, in which both parties meet directly for exchange process, and indirect finance, which involves some kind of intermediary.

Stock market. Stocks are instruments that indicate and represent ownership of the company. Stocks are divided into common and preferred. Holders of preferred stocks have a higher number of earnings compare to common stockholders; in addition, preferred stocks grant priority during events such as bankruptcy and liquidation. Both types provide a voting opportunity at shareholders' meetings.

After a company earns profit, owners may decide either to reinvest money back to the company or withdraw any amount. Companies can pay back to shareholders with buying back stocks or pay dividends. In case when a company pays dividend, it decreases the capital according to the amount paid out.

The stock market is a public institution for trading companies’ stocks. The stock value cannot be called "stable" as it is in constant change owing to many reasons. Price swings generally reflect events that affect company or market, actual value of the company and expected future value. Stock market allows companies to attract investments as well as gives a chance for investors to find a suitable project. Major stock exchange markets are New York Stock Exchange (USA), NASDAQ (USA) and London Stock Exchange Group (UK).

Bond market. Bond is a debt investment, mostly used by governments, municipalities and companies. Conceptually, an entity borrows certain amount of money for a period of time (long-term) at a fixed interest rate. Bonds are considered an alternative to bank loan. Bond markets follow the same rules as other financial markets but is oriented on long-term deals with low return and risk.

Forex. Market which provides trading of different currencies. Forex is an open market where usual participants are banks, financial organizations, investors and sole traders. This is one of the largest financial markets in the World with more than 1 trillion dollars in daily trading volume.

Money Market. Money markets (MM) allow traders exchange financial instruments (such as cash, deposit, treasury bills, banker's acceptances) with high liquidity and short-term maturity. Participants on money market are usually trading companies, banks, retailers and dealers focused on borrowing or lending. The money market plays a supporting role to financial industry as it offers short-term trading operations and rapid loans. MM are closely connected to capital markets as interest rates influence long-term interest rates of the capital market.

Investments and Trading. The goal of this part is to clarify the difference between two financial processes. Investments is a purchase of an asset or item with the idea of gaining income in long-term future. With the time moving forward, the assets' value increases, thus creating perspective wealth. Stocks and bonds are most common financial products used for investment purposes. Production of goods may also be termed an investment.

On the other hand, the goal of trading (sometimes called speculation) is to raise abnormal profits on market inefficiencies in short time spans. Speculation is hard to quantify and categorise, hence it presents high risk level. For this reason, market participants often choose investing as more attractive alternative.

2.1.2 Efficient Market Hypothesis

Efficient Market Hypothesis (EMH) was introduced by Professor Eugene Fama in 1960s. The hypothesis states that market is informatically effective as all relevant information reflected in price of asset. In other words, it is impossible to beat the market as the price already contains all information. The emergence of the theory was a statement that prices follow a "random walk", so they may not be predicted by means of technical analysis of past data.

There are different views on relative efficiency of financial market, hence practitioners usually distinguish 3 forms of EMH.

- Weak-form efficiency. Weak form of EMH states that current price of the assets already reflects all public information from the past (previous price and available data). In this case, future directions and excess return cannot be determined by technical analysis.
- Semi-strong efficiency. Compared to weak-form efficient, price not only represents past information but uses currently available market and non-market public information. There are a wide range of events that can affect the price, for example news, internet, financial reports, analytical forecast etc. Excess return cannot be achieved by fundamental analysis.
- Strong efficiency. Price instantly reflects all market, non-market and "inside" information. Strong level of efficiency is considered a "perfect market" as it reflects a situation when private information cannot to be used for personal benefits.

Criticism. The major problem with the EMH is that it assumes that all market participants arrive at a rational expectation forecast. Seller would expect a fall of the price while buyer expects rising in price. Because of that, price of asset may not only contain the true value but also reflect psychological state of the market. One of the common economical phenomena is irrationality of human behaviour in situations when one makes economical decision. This irrationality leads to market anomalies e.g. economic bubble which is a good example how market can be driven by speculative operations. Bubble refers to fast asset price escalation with investors buying larger number of products, and then quick sharp price drop. Economic bubbles are rare events; most famous ones are dot-com bubble and 2007-2008 housing bubble.

2.1.3 Modern Portfolio Theory

Modern Portfolio Theory (MPT) was introduced by Harry Markowitz in 1952. The theory attempts to establish a portfolio of assets with the aim of maximizing expected return and minimizing the level of risk. MPT suggests that investors will likely prefer portfolio with lesser risk, if both portfolios offer same expected return, meaning they will take on increased risk only with higher return. Investors evaluate risk depending on their individual risk aversion characteristics.

Return can be considered as capital appreciation of asset and dividends, for debts it may include interest payment or payment of principle. The expected return is calculated as a measured sum of the individual assets' returns. Expected return is usually based on past performance which means forecast cannot be certain. The formula for the expected return of a portfolio is:

$$E(R_p) = \sum_i w_i E(R_i)\,. \qquad (1)$$

The portfolio's risk is a complicated function of the variances of each asset and the correlations of each pair of assets. In order to measure the risk of portfolio assets' variances and their pairwise covariance are needed.

Risk is strongly correlated to expected return, the higher risk, the higher return and vice versa. At the same time, risk can be reduced by holding diversified portfolio of assets. In addition, risk can be divided into systematic and specific. Systematic risk could be managed by using short and long-term position in one portfolio. Specific risk is tied to individual assets, those can be affected by external factors such as sudden news or new governmental regulations. The formula for the expected risk of a portfolio is:

$$\sigma_p^2 = \sum_i w_i^2 \sigma_i^2 + \sum_i \sum_{j \neq i} w_i \sigma_i \, w_j \sigma_j \rho_{ij} \, . \qquad (2)$$

MPT is based on two main points: investors will always look for a higher number of return for any level of risk; level of risk can be reduced by assembling varied portfolio of unrelated assets. As large number of individuals holds risky assets in identical proportions to each other – risky assets and expected return ratios are adjusted to the ratio in which risky assets supplied to the market.

Criticism. There is an ongoing debate whether MPT is a perfect financial instrument as it is not always applicable in practice. MPT based on the expected result which means forecast of future using analysis tools; in practice, some unexpected circumstances which never appear before in historical data are simply not taken into account. MPT effort to model the probability of losses looking back to past, but it does not clarify real reasons why losses may take place. Mathematical risk measurements are only helpful as long as they represent investors' true concerns. In reality, investors are only worried about the risk as the potential loss.

2.1.4 Semivariance

Semivariance is a downside risk measure of dispersion of all observations falling below the target value in a data set. It quantifies the financial risk associated with losses – in other words, semivariance reflects the probability of the observed return being below the expected return and uncertainty about the degree of that difference. (Porter 1974; Markowitz 1991).

One of the main shortcomings of the model introduced by Markowitz (1968) lies in the assumption of symmetry and normality of the underlying return distributions. However, as already mentioned, empirical evidence seriously questions such a hypothesis.

Even minor change in return or risk estimates leads to a vastly different weight allocations in mean-variance framework (Chopra & Ziemba 1993; Ceria & Stubbs 2016). The efficient frontier computed via Markowitz' model usually overestimates the expected returns of portfolios (Broadie 1993). Ceria & Stubbs (2016) cite Michaud (1989) while referring to the 'error-maximization effect' and provide an intuitive example on how small estimation errors influence the final allocation in MPT framework. Simply speaking, if estimations are not precise, then the optimization procedure will fail to produce proper allocations.

The other flaw of the MPT lies in the risk measure itself. The MV hypothesis simply assumes that a variance of returns is a correct risk indicator. According to the model, main investor's concern is a volatility of a certain asset. That is, no matter the direction of the fluctuation, the higher the volatility – the greater the risk. Sharpe (1964) explicitly states that "under certain conditions the mean-variance (MV) model can be shown to lead to unsatisfactory predictions of investor's behaviour".

It has been recognized that investors usually do not perceive as risky those returns *above* the minimum they must earn in order to achieve their investment goals. To put it differently, the bad outcome happens when the observed return happens to be *below* some value and thus presents risk.

When the return is *above* this particular value, the individual does not view this fluctuation as risky. Estrada remarks that investors only dislike downside volatility; they do not shy away from stocks that experience large and frequent jumps above the mean; they avoid stocks that exhibit large and frequent fluctuations below the mean. Individuals are not afraid of obtaining more than their minimum acceptable return - they are afraid of obtaining less. (Estrada 2002, 2007.)

The use of downside risk measurements like *semivariance* allows to solve some problems of the original framework (Estrada 2006). In fact, Markowitz himself makes a comment on superiority of semivariance as a risk quantity. He states that analysis based on semivariance tend to produce better portfolios than those based on variance (Markowitz 1968, p194). In the revised edition of his book (Markowitz 1991) Markowitz goes as far as to claim that "semivariance is the more plausible measure of risk".

Unfortunately, there were certain difficulties associated with using mean-semivariance in practice. Computational complexity of the calculations did not allow for widespread use of the technique back at time when Markowitz carried the research. On top of that, Markowitz' hypothesis was pretty popular and well-studied piece of theory, a strong theoretical basis for future research (Tobin 1958; Hicks 1962; Sharpe 1963; 1964; Pogue 1970; Ledoit & Wolfe 2003; etc.). As for the semivariance, it only started to get attention later on in context of downside risk measures (Harlow 1991; Sortino & Van Der Meer 1991; Sortino & Price 1994).

For the purposes of the following paper the mean-semivariance (MS) optimization approach proposed by Estrada (2007b) is adapted. In a series of articles (Estrada 2002; 2006; 2007a) the author builds theoretical framework based on downside risk. He introduces alternative behavioural hypothesis based on downside risk approach as well as alternative pricing model - D-CAPM. He also makes a case for the alternative measure of risk for diversified investors (the *downside beta*) and, most importantly, suggests a heuristic optimization method in semivariance framework.

The mentioned method is of special interest for a number of reasons as it deals with all the inconveniences faced by previous researchers. First, it is very simple as it greatly simplifies the calculation problems. Second, it is fairly intuitive. Lastly, it provides a good level of approximation accuracy.

Basics. Assume an asset $\boldsymbol{i}$ with a series of returns $\boldsymbol{R_i}$, where $\boldsymbol{R_{it}}$ represents a return at time $\boldsymbol{t}$. Remember that the *variance* of this asset's returns is given by

$$\sigma^2 = E[(R_i - \mu_i)^2] = (1/T) \cdot \sum_{\boldsymbol{t=1}}^{\boldsymbol{T}} (R_{it} - \mu_i)^2 , \qquad (3)$$

where $\boldsymbol{T}$ denotes the number of observations and $\boldsymbol{\mu_i}$ is the mean return. The *covariance* between two assets $\boldsymbol{i}$ and $\boldsymbol{j}$ is then

$$\sigma_{ij} = E[(R_i - \mu_i)(R_j - \mu_j)] = (1/T) \cdot \sum_{t=1}^{T} (R_i - \mu_i)(R_j - \mu_j) . \qquad (4)$$

The *semivariance* of asset $\boldsymbol{i}$'s return with respect to a benchmark $\boldsymbol{B}$ ($\boldsymbol{\Sigma_{iB}^2}$)is defined as

$$\Sigma_{iB}^{\ 2} = E\{[Min(R_i - B, 0)]^2\} = (1/T) \cdot \sum_{t=1}^{T} [Min(R_{it} - B, 0)]^{\boldsymbol{2}} , \qquad (5)$$

where $\boldsymbol{B}$ is any benchmark return chosen by investor. The square root of (5) is the *semi-deviation* of asset $\boldsymbol{i}$ with respect to a benchmark $\boldsymbol{B}$.

The *semi-covariance* between assets $\boldsymbol{i}$ and $\boldsymbol{j}$ ($\boldsymbol{\Sigma_{ij}}$)with respect to a benchmark $\boldsymbol{B}$, as defined by Estrada (2007), is

$$\Sigma_{ijB} = E\{[Min(R_i - B, 0)] \cdot [Min(R_j - B, 0)]\} , \qquad (6)$$

or, equivalently, can be expressed as

$$\Sigma_{ijB} = (1/T) \cdot \sum_{t=1}^{T} [Min(R_{it} - B, 0) \cdot Min(R_{jt} - B, 0)] . \qquad (7)$$

Such a definition allows to compute for any desired $\boldsymbol{B}$ and generate a *symmetric* ($\boldsymbol{\Sigma_{ijB} = \Sigma_{jiB}}$) and *exogenous* matrix.

Symmetry simply means that after the *transposition* (switching the row and column indices of the matrix) the "new" transposed matrix is the same as the "old", pre-transposed version. By definition of symmetry the matrix has equal number of dimensions as well (said to be *quadratic*).

An *exogenous variable* is “a factor in a causal model or causal system whose value is independent from the states of other variables in the system” (Bryman, Lewis-Back, Liao 2004). Thus, matrix is said to be exogenous if each variable stored inside is independent of the values of other variables.

Finally, the expected return $\boldsymbol{E_p}$ and variance $\boldsymbol{\sigma^2}$ of a portfolio are given by

$$E_p = \sum_{i=1}^{n} x_i E_i , \qquad (8)$$

$$\sigma^2 = \sum_{i=1}^{n} \sum_{j=1}^{n} x_i x_j \sigma_{ij} , \qquad (9)$$

where $\boldsymbol{x_i}$ represents the proportion of the portfolio invested in asset $\boldsymbol{i}$, $\boldsymbol{E_i}$ is the expected return of asset $\boldsymbol{i}$, and $\boldsymbol{n}$ is the number of assets in the portfolio.

Estrada argues that the semivariance of the portfolio with respect to a benchmark can be approximated with

$$\Sigma_{pB}{}^2 \approx \sum_{i=1}^{n} \sum_{j=1}^{n} x_i x_j \Sigma_{ijB} , \qquad (10)$$

where $\boldsymbol{\Sigma_{ijB}}$ is defined as in (7).

Note that the expression above approximates true semi-covariance matrix rather than computes it explicitly. There is a different definition of semi-covariance matrix given by Markowitz (1968) which, indeed, provides us with an exact result. The main problem of the latter formula lies in *endogeneity* of the resulting matrix: its elements (pairwise assets’ semi-covariance) depend on whether *a portfolio* underperforms the benchmark which, in turn, depends on *the weights* of the assets in such a portfolio, forming a circular dependence of a kind.

In this way, in order to find optimal portfolio using conventional methods one has to go through the following process: first, compute a set of all feasible portfolios, second, from their returns calculate exact semi-deviations and finally choose the one with the lowest value. However, to find a truly optimal solution this set has to contain every possible combination of assets, which is computationally intractable.

Instead, Estrada suggests estimating the matrix in a way that would not depend on portfolio performance. Take note that with Markowitz' (1959) definition one has to know if *a portfolio* underperforms the benchmark. In turn, with (7) one has to know if *an asset* performs less well than the benchmark.

Recall that the expression (10) returns a symmetric and exogenous semi-covariance matrix, which then could replace the original covariance matrix in the solution of mean-variance problems. There are several possible descriptions of such tasks which depend on the particular investor: some may aim to minimize risk, others may aim to minimize risk subject to a target return or maximize return subject to a target level of risk. The solution proposed by Estrada is suited for all kinds of definitions.

For the sake of example consider the problem of maximizing risk-adjusted return. Substituting for semi-covariance matrix estimated using (10) the optimization objective becomes:

$$\textit{Minimize} \qquad \sum_{i=1}^{n} \sum_{j=1}^{n} x_i x_j \Sigma_{ijB} - \sum_{i=1}^{n} x_i \mu_i \qquad (11)$$

$$\textit{subject to} \qquad \sum_{i=1}^{n} x_i = 1 ,$$

$$x_i \geq 0 , \qquad (12)$$

where $\boldsymbol{\Sigma_{ijB}}$ is a semi-covariance between asset $\boldsymbol{i}$ and $\boldsymbol{j}$ with respect to a benchmark $\boldsymbol{B}$ as in (7). Notice the constraints: all weights of assets in a portfolio must be non-negative (greater than or equal to zero) and sum to one. This way, short-selling of a security is restricted to protect from additional risk and uncontrollable losses.

This form of expression allows us to minimize expected risk while simultaneously maximizing the expected return. For this expression to be minimal, the risk measurement should be as small as possible and return measurement should be as large as possible.

In a **vector form,** one is presented with a general quadratic expression of a kind

$$\textit{Minimize} \qquad x^T \Sigma\, x - q * R^T x \qquad (13)$$

$$\textit{subject to} \qquad x^T 1 = 1$$

$$x \geq 0\,, \qquad (14)$$

where once again $\boldsymbol{\Sigma}$ is a semi-covariance matrix of returns estimated with (10), $\boldsymbol{x^T}$ and $\boldsymbol{R^T}$ are transposed column vectors of weights and returns respectively, $\boldsymbol{x^T 1}$ is a dot product of column vector of weights and row vector of ones.

An Assessment. Although an estimate, it is a reasonably close one. Estrada puts his model through various evaluation routines and comes with an assuring evidence.

In order to test the accuracy, exact and approximate semi-deviations were calculated for over 1,100 portfolios, some containing stocks, some – markets, and some – other asset classes. While comparing true and approximate values of semi-deviations the author observes high levels of correlation between them. Further, in all cases when the approximation errs it does so on the side of caution by overestimating the true risk of a portfolio. (Estrada 2007, 13-18.)

Despite the fact that outcomes are persuasive, one has to accept certain imperfections that come with the adopted approach and account for the possibility of moderate errors in computations. Taking a closer look at the results concerning emerging markets (EMs) and DJIA stocks (Estrada 2007, 14, Exhibit 4, Panel B & C) one might point out the magnitude of a difference between true and estimated portfolio semi-deviations: it ranges from 0.68 to 2.10 percent for EMs and 2.07 to 2.67 percent for DJIA stocks. These errors are quite large even for annual values of an asset. Moreover, a keen reader would probably question the representativeness of the sample as, firstly, 30 securities is considered to be the least amount to form a properly diversified portfolio (Statman 1987) and, secondly, DJIA itself might be quite biased in depicting the dynamics of stock market (Mueller, Padmaraj, St-John 1999; Cerin & Dobers 2001; Platt, Cai, Platt 2014).

It might be tempting to compare the performance of a classic MV framework with SM. Doing so is senseless to a degree. By definition, the objective of MV optimizer is to maximize the expected return per unit of volatility, while MS will maximize the expected return per unit of volatility *below the chosen benchmark*. “In the end, it all comes down to what any given investor perceives as the more appropriate measure of risk” (Estrada 2007, 18).

2.1.5 Shrinkage Estimators and Bayes-Stein

In statistics, an *estimator* is a rule for calculating an estimate of a given quantity based on observed data. Think of it as a *function* employed to infer or guess the value of an unknown parameter in a statistical model. As a matter of fact, there is no restrictions on which functions of the data can be called “estimators”.

A *shrinkage estimator* is the one that, either explicitly or implicitly, applies the effects of shrinkage - reduction in distance of some sort. The term “shrinkage” indicates that the transformed estimate is made closer to some predetermined value than the raw estimate. Simply put, this implies that a naive or raw approximation is enhanced by combining it with some other information. An interesting fact is that many standard estimators can be enhanced, in terms of mean squared error (MSE), by shrinking them towards some fixed constant value. The performance of this new modified estimator is sometimes better, *but never worse* than that of an original one.

Application in finance. In context of portfolio theory and optimization vectors of future returns of a security are of special interest. There is no acknowledged way to predict these exactly, so one has to be content with various approximations. However, an investor usually wants forecasts to be as accurate as possible. Coupled together, these facts present a considerable challenge for anyone interested in portfolio-related problems as they introduce uncertainty and additional risk of losses when being wrong on a guess. This estimation risk stems from the fact that roughly calculated returns might simply not contain any useful information about the asset as they did not capture true moments of underlying distribution, thus rendering optimization pointless.

Ceria and Stubbs emphasize that most of the estimation risk in optimal portfolios come from errors in approximations of expected returns more so than those of risk (Frankfurter, Phillips & Seagle 1971; Dickinson 1979; Jobson, Korkie & Ratti 1979; Ceria & Stubbs 2006, 1-2). Further, the authors revisit possible approaches to tackling such a task (Black & Litterman 1991; Michaud 1999; Horst, de Roon, Werker 2002; Cavadini, Sbuelz, Trojani 2001).

One of the more common techniques mentioned is the utilization of James-Stein estimators (Jobson & Korkie 1980). These methods allow shrinking prospective returns towards the average expected return based on the volatility of an asset and the distance of its expected return from the average. Jorion (1986) developed a similar technique that brings future return estimates closer towards the minimum variance portfolio.

In this paper, the route of shrinkage estimators was taken for a few simple reasons. First, these techniques are fairly intuitive - the reader will be able to get the idea of the process without knowing complex statistical theory and mathematical derivations behind those concepts. Secondly, the techniques themselves are quite simple in a computational sense relative to the existing alternatives (Efron & Morris 1975; Scherer 2002, Ceria & Stubbs 2006). Lastly, shrinkage estimators are directly applicable to our mean-semivariance framework as they adjust the estimates of expected return directly and do not interfere with either semi-covariance matrix computation procedure or actual optimization routine.

While considerable efforts have been made to correct approximations of expected returns, there will always be errors in these estimates because of the inherent random nature of the asset return process. Even those employing Bayesian procedures such as James-Stein or Black-Litterman methods admit that estimation error remains a factor in the enhanced estimates of expected returns, even if it is significantly less than that obtained without the use of these methods (Ceria & Stubbs 2006).

James-Stein Paradox. The “James-Stein Paradox” paper by Efron & Morris (1997) serves as a respectable introductory example into the world of shrinkage estimators. Their work thoroughly discusses properties of James-Stein estimators and gives a solid understanding without rigorous math and many technical.

Stein's paradox concerns the use of observed averages to estimate underlying moments of distribution. Consider a baseball example from the paper. A baseball player who gets 7 hits in 20 official times at bat is said to have a batting average of .350 ($.350 == 7/20$). By computing this statistic, an estimate of the player's true batting ability in terms of his observed average rate of success is constructed. Asked how well the player will do in his next 100 trials, one would probably predict 35 more hits. In traditional statistics, it can be proved that no other estimation rule is uniformly better than the observed average.

The paradoxical element in Stein's result is that it sometimes contradicts the elementary law of statistical theory stated above. If one has three or more baseball players, and if one is interested in predicting future batting averages for each of them, then there is a procedure that is more efficient than simply generalizing from three separate simple averages. The statistician who employs Stein's method can expect to predict the future averages more accurately *no matter* what the true batting abilities of the players may be.

The example considered in the article relates to the batting averages of baseball players. Stein used the shrinking adjustment procedure in order to get new estimates for player’s true batting averages for the following season. For 16 out 18 players the James-Stein adjusted average turned out to be more accurate than “simple” average; when later comparing with the results over the whole season, adjusted averages turned out to be closer to the “true”, or seasonal, averages than the counterpart. By applying total-squared error as a comparison metrics Stein showed that his method was 3.5 times more accurate than commonly used simple means estimation.

Next, the authors offer an interesting thought experiment. To the pool of 18 variables (player's batting averages) the 19th was added - the proportion of imported Cars in Chicago. While being somewhat unreasonable, the entire procedure happened to work just fine. The theorem applies as well to the new 19 random variables as it did to the 18 original ones. The same confusing difference could be displayed another way: why should a success or lack of success of one player influence the estimation for another player?

To understand why shrinkage behaves better than simple maximum likelihood estimation let us get back to taking averages as a statistical procedure. It might not be that obvious why the average is so often used in estimating the central tendency of an unknown parameter. The explanation lies in the distribution of random variables.

In statistics, the most common distribution is the "normal" one, described by bell-shaped curve. It was first studied by Gauss and it is specified by the use of 2 parameters: the **mean** (central tendency or the most typical value) and the **standard deviation** (the variation or dispersion of observations). In practice, one often has to infer the mean and the standard deviation of the underlying distribution from the collection of observed values.

Theoretically, mean can take any value, but some are more likely than others. Gauss showed that among all possible choices the average of the observed data *maximizes* the probability of obtaining the underlying mean. Further, Fischer proved that all the information about the mean that could be possibly found in data is contained in the average of the observed values. (Efron & Morris 1975). In 1950s Blyth, Lehmann and Hodges proved that the average is *admissible* when it is applied to one series of observations for the purpose of estimating unknown mean (Blyth 1951; Hodges & Lehmann 1951). In other words, for a set of observations of one random variable no better estimator exists.

Stein's theorem applies to estimating several unknown means. James-Stein's paradox is simply their proof (Stein 1956; James & Stein 1961) that when the number of unknown means happens to be more than two, there exists a better estimator than simple averages of those series. James and Stein not only proved the existence of such estimators but also provided an example.

James-Stein estimator is defined as

$$z = \underline{y} + c(y - \underline{y}), \qquad (15)$$

where $\boldsymbol{y}$ is the average of a single set of observations, $\underline{\boldsymbol{y}}$ is the average of averages and $\boldsymbol{c}$ is the shrinking factor

$$c = 1 - \frac{(k-3)\sigma^2}{\Sigma(y-\underline{y})^2}. \qquad (16)$$

The $\boldsymbol{k}$ is the number of unknown means and σ^2 is the common variance.

The procedure makes a guess that all the unobservable means are near the certain value $\underline{\boldsymbol{y}}$. If the data supports that guess, the estimates are shrunk further towards the $\underline{\boldsymbol{y}}$. If the assumption does not hold, the magnitude of shrinkage is less drastic. Calculated this manner, the precision of James-Stein estimator is more than that of the sample averages *irrespective of* what the true values of the means end up being.

The accuracy is greatest when all the means come near the same value and gradually decreases as they depart from each other. The surprising finding is that this accuracy is *always* better than or equal to the one of simple averaging approach. Even in cases when the procedure does not significantly increase the accuracy, there is little penalty for using it; shrinkage cannot produce larger total mean squared error than the maximum-likelihood estimation. The rules discussed above are also robust to the assumption of the normal distribution. (Efron & Morris 1975.)

It is important to understand the limitations of the James-Stein procedure. When the means have unconventional values, James-Stein estimator is not guaranteed to work. In fact, it can create serious errors and degrade the estimation of seriously atypical mean. Recall cars and baseball statistics example. Now one can see why the idea of adding them together is a bad decision: there exists considerable probability that the automobiles would be irregular. When estimating the mean of just one random variable from the observed series of values or when the observed random values have little to do with each other, one is better off using simple average. (Efron & Morris 1977.)

Jorion's Bays-Stein estimator. Jorion (1986) presents an application of shrinkage to portfolio selection problems. It provides the necessary theoretical basis and illustrates the extent of possible gains over classical estimators. Jorion builds upon previous studies by Barry (1974), Brown (1976) and Klein & Bawa (1976). He puts the estimation procedure in the portfolio context, where the main objective is to minimize the impact of estimation risk on optimal portfolio choice. Further, Jorion proposes a shrinkage estimator which brings the means towards the common value; this leads to decreased estimation error with more than two securities in a portfolio. The effect of estimation error for all assets is summarized into one loss function, which is minimized as a whole.

In a one-period model, investor usually wants to maximize the expected utility of his or her end-of-period wealth. In terms of rates of return, the task is to choose a set of weights $\boldsymbol{q}$ in order to maximize the expected utility $\boldsymbol{U}(\boldsymbol{z})$ of return on the portfolio $\boldsymbol{z} = \underline{\boldsymbol{q}}^{\boldsymbol{T}}\underline{\boldsymbol{r}}$, where $\boldsymbol{r}$ is the vector of future observations,

$$EU(z) = \int U(z)p(z \mid \underline{\theta})dz\,, \qquad (17)$$

subject to a feasibility constraints. It contains a utility function $\boldsymbol{U}(\boldsymbol{z})$, which can be different across investors, and the conditional distribution of rates of return $\boldsymbol{p}(\boldsymbol{z} \mid \underline{\boldsymbol{\theta}})$, dependent on a set of parameters $\underline{\boldsymbol{\theta}}$, unknown for all practical purposes.

In a *certainty equivalent* model, where simple averages are believed to be equivalent to the true parameters, one *assumes* that true underlying parameters $\underline{\boldsymbol{\theta}}$ are equal to their estimated ones $\underline{\boldsymbol{\theta}}(\underline{\boldsymbol{y}})$, based on some estimator defined as a function of the observations $\underline{\boldsymbol{y}}$. Thus, the optimization objective is to maximize the expected utility given that the estimated parameters are same as their true counterparts:

$$\textit{Maximize} \qquad E_{\underline{y}}[U(z) \mid \underline{\theta} = \hat{\theta}(\underline{y})]\,. \qquad (18)$$

This approach clearly disregards the problem of estimation risk, or parameter uncertainty; by definition it does not care how accurate the estimates are. The Bayesian solution is to express this uncertainty in terms of the *predictive density function* (Zellner & Chetty 1965). Such a function describes the distribution of possible unobserved values conditional on the observed values. To simplify, it allows to account for estimation error explicitly and thus minimize it.

If the parameters of the returns distribution are known, it is trivial to express a utility function and optimize it (Jorion 1986, 282). On the other hand, if the parameters are unknown, which is the case in finance, the choice would be made on a basis of some estimate. Therefore, the choice would be non-optimal. The value of utility function based on estimates, however precise, will necessarily be lower than the value of one based on true underlying moments. Such a loss in utility can be expressed as a function of the data.

Jorion shows (1986, 285) that the predictive density function $\boldsymbol{p}(\underline{\boldsymbol{r}} \mid \underline{\boldsymbol{y}}, \boldsymbol{\Sigma}, \boldsymbol{\lambda})$ of the future returns vector, conditional on $\boldsymbol{\Sigma}$ and $\boldsymbol{\lambda}$, is multivariate normal, with *mean*

$$E[\underline{r}] = (1 - w)\underline{Y} + w\underline{1}Y_0\,, \qquad (19)$$

where

$$w = \frac{\lambda}{T + \lambda}, \qquad (20)$$

$$Y_0 = \frac{\underline{1}^T \Sigma^{-1}}{\underline{1}^T \Sigma^{-1} \underline{1}} \underline{Y}\,, \qquad (21)$$

and covariance matrix

$$V[r] = \Sigma(1 + \frac{1}{T + \lambda}) + \frac{\lambda}{T(T + 1 + \lambda)} \frac{\underline{1}^T \underline{1}}{\underline{1}^T \Sigma^{-1} \underline{1}}, \qquad (22)$$

where $\boldsymbol{T}$ is the number of observations, or time periods, $\boldsymbol{w}$ is the shrinkage coefficient, $\underline{\boldsymbol{Y}}$ is the vector of observed averages, $\boldsymbol{\Sigma^{-1}}$ is the inverse of the sample covariance matrix, $\underline{\mathbf{1}}$ is the column vector of ones.

For more details and rigorous derivations, the reader is referred to Jorion's original paper (1986, appendix). An interesting observation is that the grand mean $\boldsymbol{Y_0}$ happens to be the average return for the minimum variance portfolio.

The richness of the Bayes approach is that $\boldsymbol{\lambda}$ is estimated directly from the data. The PDF $\boldsymbol{p}$ ($\boldsymbol{\lambda} \mid \underline{\boldsymbol{\mu}}, \boldsymbol{\eta}, \boldsymbol{\Sigma}$) is a *gamma distribution* with mean $\boldsymbol{N + 2 / d}$, where $\boldsymbol{d}$ is defined as $(\underline{\boldsymbol{\mu}} - \underline{\mathbf{1}}\,\boldsymbol{\eta})^{\boldsymbol{T}}\,\boldsymbol{\Sigma}^{-1}\,(\underline{\boldsymbol{\mu}} - \underline{\mathbf{1}}\,\boldsymbol{\eta})$ and is replaced by its sample estimate $(\underline{\boldsymbol{Y}} - \underline{\mathbf{1}}\,\boldsymbol{Y_0})^{\boldsymbol{T}}\,\boldsymbol{\Sigma}^{-1}\,(\underline{\boldsymbol{Y}} - \underline{\mathbf{1}}\,\boldsymbol{Y_0})$. The shrinkage coefficient is then

$$w = \frac{N + 2}{(N + 2) + (\underline{Y} - \underline{1}\,Y_0)^T\, T\Sigma^{-1}\,(\underline{Y} - \underline{1}\,Y_0)}, \qquad (23)$$

where $\boldsymbol{N}$ is the number of assets in a portfolio.

In practice, $\boldsymbol{\Sigma}$ is said to be unknown, and the author suggests replacing it, as in Zellner & Chetty (1965), with

$$\Sigma = \frac{T - 1}{T - N - 2} S, \qquad (24)$$

where $\boldsymbol{S}$ is the usual unbiased sample covariance matrix.

An Assessment. The performance of various estimators is measured by the loss of utility due to estimation error, averaged over repeated samples. Since the risk function is intractable in this particular case, Jorion resorts to simulation analysis.

He uses sample estimates from stock market returns for seven major countries, calculated over 60-month period. Observe that standard deviations are not too different across assets, although they are somehow large relative to sample means (Jorion 1986, 287, table 1).

For each drawing $\boldsymbol{k}$, the optimal portfolio was computed for each possible estimator tested, leading to different values of the derived expected utility. The experiment was repeated **K** = 1000 independent times and the risk function was defined as the average loss of expected utility. To account for effect of sample size, the previous operations were repeated for various time periods **T** ranging from 25 to 200.

Bayes-Stein estimator is shown to always have lower risk than both the certainty equivalence and the Bayes diffuse prior estimators. The improvement is noticeable and significant: the measured difference ranges from 8 to 0.2 percent per annum. Bayes-Stein is shown to always outperform the sample mean no matter the true parameter value. Jorion notes that the test results might still provide conservative estimates of gains of the former procedure. In studies conducted further, he evaluated the out-of-sample performance of various estimators, based on actual stock return data, and found that shrinkage significantly outperformed the classical sample means.

2.1.6 Portfolio Return Measurements

Although expected levels of risk and return are already fairly informative indicators of a portfolio performance, potential investors would still like to know more. For this reason, number of other measurements were introduced to study the portfolio in detail and make various evaluations. Different performance measures allow to compare the performance of different portfolios as well as better understand the dynamics. The following chapter includes some widely used indicators and ratios as well as some less common types.

Table 2.1.1: Performance Measures

Measure	Description
Expected Return	Mean of a return series
Variance	Variance of a return series
Alpha	Excess abnormal return earned over a market
Beta	Relative volatility of a portfolio to a market
Sharpe ratio	The ratio of excess returns over risk-free asset adjusted for *portfolio* risk
Treynor measure	The ratio of excess returns over risk-free asset per unit of *market* risk
Jensen's alpha	Excess abnormal return over the *theoretical expected return*
Sortino ratio	The ratio of excess returns over specified target return adjusted for *downside portfolio* risk
Max DD	Maximum Drawdown is the difference between the highest and the lowest value the portfolio reached over a time period

Rates of return. In finance, a return is a gain or loss on an investment. It involves any change in value including interest, dividends and other such cash flows. The return could be measured either in absolute terms (e.g., euros) or as a percentage of the initial amount invested. A loss is described as a *negative return*.

Rate of return is a profit on an investment over a period of time, expressed as a proportion of the original investment. The return over a single period is:

$$r = \frac{V_1 - V_0}{V_0}, \qquad (25)$$

where V_0 is the initial investment and V_1 is the value at the end of a time period. The time period is typically a year, in which case the rate of return is referred to as annual return.

In order to compare returns over time periods of different lengths on an equal basis, it is useful to convert each return into an annual equivalent rate of return, or *annualised return*. This conversion process is called annualization:

$$R_{annual} = (r_{cummulative} + 1)^{365/_{days\ held}} - 1 \qquad (26)$$

For example, given the monthly return of 3% the annualized returns are then $(0.03 + 1)^{12} - 1 = 43\%$.

Alpha and Beta. Alpha is a measure of the return of an investment compared to the chosen market index. An alpha of 5% means the return on investment over a selected period of time was 5% better than the benchmark during that same period; a negative alpha signifies the investment behaved worse than the index.

The ones using alpha in measuring the performance usually assume that a portfolio of interest is already diversified enough to eliminate unsystematic, company- or industry-specific risk. Because alpha demonstrates the relative performance of a portfolio, it is often considered to represent, as in our case, the value that a particular model adds to or subtracts from investor's return. To rephrase, alpha is the return on an investment that is not a result of general movement in the greater financial market.

In efficient markets, the alpha value is exactly zero. Therefore, the alpha coefficient indicates how an investment has performed after accounting for the risk it involved. If the alpha is less than zero, the investment earned too little for the risk taken. If the alpha is more than zero, the investment earned return on top of the adequate reward for the assumed risk. An alpha of 0 would indicate that the portfolio is tracking the benchmark perfectly and that there is no added or lost value.

Beta (β or beta coefficient) is a measure of the relative volatility, or systematic risk; it shows whether the investment is more or less volatile than the market as a whole. Usually a beta less than 1 is a sign that the investment is less volatile than the market, while a beta more than 1 states that the investment is more volatile than the market. This variability is measured in the form of standard deviation. It is used in the CAPM to calculate the return premium of an asset.

Beta is a measure of the risk arising from exposure to general market movements as opposed to idiosyncratic, local factors. The portfolio of all the assets investable has a beta of exactly 1 and is called market portfolio. It is important to understand that a beta is computed through a regression and therefore is a measure of correlation. As such, beta close to zero can indicate either an investment with lower variability than the market, or a volatile investment whose price movements are not highly correlated with the market. An example of the first is a US treasury bill: the price does not normally experience fluctuations, hence a low beta. An example of the second is gold. The price of gold is substantially volatile, but not in the same direction or at the same time as the one of the market portfolio.

A beta greater than one generally signifies that the asset's price is rather volatile and tends to follow the movement of the market. An example is a stock in a technology company. Negative betas are possible for investments that somehow oppose market's movement: these go up when the market goes down, and vice versa. There are few fundamental investments with stable and significant negative betas, but some derivatives can have large negative betas.

Beta is crucial because it measures the risk that cannot be reduced by means of diversification. It does not measure the risk of an investment held on a stand-alone basis, but the amount of risk the investment adds to an already-diversified portfolio. In the CAPM, beta is the only kind of risk for which investors should receive an expected return higher than the risk-free rate of interest.

The **beta** and **alpha** coefficients are essentially parameters in the CAPM. Recall the formula of an expected return of a security i:

$$E(R_i) = R_f + \beta_{iM}(E(R_m) - R_f), \qquad (27)$$

where R_f is a risk-free rate, β_{iM} is a beta of a security and $E(R_M)$ is expected market return.

The Security Characteristics Line pretty much graphs the performance of a particular asset or portfolio against that of the market portfolio at every point in time:

$$SCL: R_{i,t} - R_f = \alpha_i + \beta_i (R_{M,t} - R_f) + \epsilon_i, \qquad (28)$$

where alpha is the intercept and beta is the slope of a fitted line.

Alphas and betas can be computed by linear regression analysis of the excess return over the risk-free one of a portfolio versus the excess returns of a benchmark portfolio.

Sharpe ratio. The Sharpe ratio (also Sharpe index, or reward-to-variability ratio) is a way to examine the relative performance of an investment by adjusting for its risk. It measures the excess average return (or risk premium) per unit of deviation in an asset or an investment model, often dubbed as risk.

The Sharpe measurement shows how well the return of compensates the investor for the risk taken. The intuition is that a portfolio with “zero risk” investment, such as US treasury bills, has a Sharpe of exactly zero. When comparing two assets or portfolios versus a common reference, the one with a higher Sharpe ratio provides better return for the same amount of risk (or the same return for lower risk). To simplify, the greater the value of Sharpe ratio, the more attractive the investment.

Since its revision by the original author, William Sharpe, in 1994, the Sharpe ratio is defined as:

$$S_i = \frac{E[R_i - R_f]}{\sigma_i}, \qquad (29)$$

where σ_i is a deviation of asset's returns.

Treynor ratio. The Treynor ratio (sometimes called reward-to-volatility ratio) is a measurement of the returns earned in excess of that which could be earned on an investment that has no diversifiable risk (e.g., US treasury bills), per unit of *market* risk.

The Treynor ratio ties excess return over the risk-free rate to the additional risk taken; however, unlike in Sharpe's method, systematic risk is used instead of total risk. By the same logic though, the higher the Treynor ratio, the better the performance of the portfolio under analysis.

Treynor ratio is defined as

$$T_i = \frac{E[R_i - R_f]}{\beta_i}. \qquad (30)$$

Jensen's alpha. In finance, Jensen's alpha (or ex-post alpha) is used to determine the abnormal return of a security or portfolio over the theoretical expected return. It is much like the standard alpha but based on a theoretically predicted performance of the index instead of a market index.

Usually such a performance, given investment's beta and the average market return, is predicted using CAPM. In this context, calculating alpha requires the following inputs:

- the realized return $\boldsymbol{R_i}$
- the market return $\boldsymbol{R_M}$
- the risk-free rate of return $\boldsymbol{R_F}$
- the beta of the portfolio $\boldsymbol{\beta_{iM}}$ relative to the market return

To better understand the Jensen's alpha, consider an expression

$$j = (R_i - R_f) - \beta_{iM}(R_M - R_f)\,. \qquad (31)$$

Sortino ratio. The Sortino ratio measures the risk-adjusted return of an asset, portfolio, or investment model. It is reminiscent of the Sharpe ratio but penalizes only those returns falling below a specified target or required rate of return, while the former penalizes both upside and downside volatility evenly. Though either ratio measures an investment's risk-adjusted return, they do so in considerably different ways that very often lead to varying conclusions as to the true nature of the investment's return-generating efficiency.

The ratio is calculated as follows:

$$S = \frac{R-T}{DR}, \qquad (32)$$

where $\boldsymbol{R}$ is the average realized return, $\boldsymbol{T}$ is the target value and $\boldsymbol{DR}$ is the downside risk.

2.2 Technology Methods

2.2.1 Mathematical Notation and Concepts

The following chapter contains a brief review of mathematical concepts and notations used throughout the paper. The assumption is that the reader is at least familiar with some of them and provide the review for reference purposes.

The paper heavily relies on linear algebra and touches upon topics such as multivariable calculus, optimization and probability theory. For further discussion of linear algebra and exhaustive details the reader is referred to Linear Algebra (Cherney, Denton, Thomas & Waldron 2013) and other math-related resources.

Vectors. Essentially, vectors are things one can add and multiply. They serve as a handy tool to summarize and express the data in a readable and understandable format. A vector $\boldsymbol{v} \in \boldsymbol{R^n}$ represents ordered series of individual variables $\boldsymbol{v_i} \in \boldsymbol{R}$. The index $\boldsymbol{i} \in \{\mathbf{1}, \cdots, \boldsymbol{n}\}$ indicates the position of a particular in a vector.

The orientation of a vector matters when performing various operations. By default, the variables are considered to be listed vertically as in (7.1) and are called *column vectors*. Equation (7.2) illustrates the **transpose** operation T, where v is flattened; the flattened vector is sometimes called *row vector*. A neat way to represent a column vector is thus $\boldsymbol{v} = [v_1 \ \cdots v_n \ ]^T$.

$$\boldsymbol{v} = \begin{bmatrix} v_1 \\ \vdots \\ v_n \end{bmatrix} \qquad (2.2.1)$$

$$\boldsymbol{v^T} = [v_1 \quad \cdots \quad v_n] \qquad (2.2.2)$$

Vectors are well characterized by their dimension, which, simply put, specifies the number of variables in a vector. For example, (2.2.1) and (2.2.2) both have n variables, thus, they are called $\boldsymbol{n}$-dimensional vectors.

To add a vector to another one, they both must satisfy a certain condition: their dimensions should be the same. Assuming equal number of dimensions, vector addition is just adding values of corresponding dimensions.

A vector can be *multiplied*, or rescaled, by a real number $\boldsymbol{r}$. To scale a vector, one has to scale each of his dimensions by that same number.

In mathematics, the *dot product* is an operation that, given two sequences of numbers of equal length, returns a single number. Algebraically, the *dot product* of two vectors (sometimes called inner product) is defined as follows:

$$a \cdot b = \sum_{i=1}^{n} a_i b_i = a_1 b_1 + a_2 b_2 + \cdots + a_n b_n. \qquad (33)$$

The intuitions behind scalar multiplication (2.2.3), vector addition (2.2.4) and inner product (2.2.5) are given below.

$$5 * \boldsymbol{v} = 5 \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} = \begin{bmatrix} 5 * 1 \\ 5 * 2 \\ 5 * 3 \end{bmatrix} \qquad (2.2.3)$$

$$\boldsymbol{a} \pm \boldsymbol{b} = \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} \pm \begin{bmatrix} 4 \\ 5 \\ 6 \end{bmatrix} = \begin{bmatrix} 1 \pm 4 \\ 2 \pm 5 \\ 3 \pm 6 \end{bmatrix} \qquad (2.2.4)$$

$$\boldsymbol{c} \cdot \boldsymbol{d} = \boldsymbol{c^T d} = \begin{bmatrix} 1 & 2 \end{bmatrix} \begin{bmatrix} 4 \\ 5 \end{bmatrix} = 1 * 4 + 2 * 5 \qquad (2.2.5)$$

Matrices and vectors have a lot in common. In fact, matrices are extensions of vectors: a matrix is a rectangular array of numbers, symbols, or expressions, arranged in *rows* and *columns*, for which operations such as *addition* and *multiplication* are defined.

As an example, consider an $\boldsymbol{m} \times \boldsymbol{n}$ matrix (2.2.6): the individual item $\boldsymbol{a_{ij}}$is called *an element* or *an entry*, where $max\ \boldsymbol{i} = \boldsymbol{m}$, $max\ \boldsymbol{j} = \boldsymbol{n}$. The size of this matrix is defined by the number of rows and columns and equals $\boldsymbol{m}\ x\ \boldsymbol{n}$; **m** and **n** are called *dimensions* of the matrix.

$$\boldsymbol{M} = \begin{bmatrix} m_{11} & \cdots & m_{1n} \\ \vdots & \ddots & \vdots \\ m_{n1} & \cdots & m_{nn} \end{bmatrix} \qquad (2.2.6)$$

To *transpose a matrix* is to turn its rows into columns and vice versa:

$$\boldsymbol{M^T} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}^{\boldsymbol{T}} = \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix} \qquad (2.2.7)$$

A matrix is composed of multiple vectors representing either rows or columns. Somewhat Pythonic notation is utilized to reference row or column vectors. For example, first row in $\boldsymbol{M}$ would be expressed as $\boldsymbol{M}[\boldsymbol{0}][:]$. Conversely, the first column would be mentioned as $\boldsymbol{M}[:][\boldsymbol{0}]$. First [ ] represents the row axis, the second – the columns axis. The index inside the brackets marks the position of a cell in a container. To obtain the range of indices a colon is used - a *slice.*

Provided that they have the same size (each of them has the same number of rows and the same number of columns), two matrices can be *added or subtracted* element by element as in (2.2.8)

$$\boldsymbol{A} \pm \boldsymbol{B} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \pm \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \begin{bmatrix} 1 \pm 5 & 2 \pm 6 \\ 3 \pm 7 & 4 \pm 8 \end{bmatrix}. \qquad (2.2.8)$$

Any matrix can be multiplied element-wise by a scalar, much like a vector. The rule for *matrix multiplication* of two matrices, or a matrix and a vector, however, is that two matrices can be multiplied only when the number of columns in the first equals the number of rows in the second (i.e., the inner dimensions are the same, **n** for $\boldsymbol{A_{m,n}} \times \boldsymbol{B_{n,p}}$).

Using the dot product rule for vectors, each row vector of matrix $\boldsymbol{A}$ is multiplied by every column vector of matrix $\boldsymbol{B}$

$$\boldsymbol{AB} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \begin{bmatrix} 1*5+2*7 & 1*6+2*8 \\ 3*5+4*7 & 3*6+4*8 \end{bmatrix} .\textit{(2.2.9)}$$

A special case of matrix multiplication is multiplication of a matrix $\boldsymbol{A} \in \boldsymbol{R^{m \times n}}$ by a vector $\boldsymbol{b} \in \boldsymbol{R^{n}}$ (7.8). The output is the matrix where an entry is a dot product of vector $\boldsymbol{b}$ and each row vector $\boldsymbol{A}$:

$$\boldsymbol{Ca} = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} = \begin{bmatrix} 1*1+2*2+3*3 \\ 4*1+5*2+6*3 \\ 7*1+8*2+9*3 \end{bmatrix} \textit{(2.2.10)}$$

Very often while working with vectors and matrices one wants to perform an operation on each element. For instance, one might want to multiply corresponding elements of two matrices. For this purpose, the *element-wise operations* are usually defined in various numerical processing environments. Observe the following expressions to see the examples.

$$\boldsymbol{A} .* \boldsymbol{B} = \begin{bmatrix} 1 & 2 \\ 3 & 4 \end{bmatrix} .* \begin{bmatrix} 5 & 6 \\ 7 & 8 \end{bmatrix} = \begin{bmatrix} 1*5 & 6*2 \\ 3*7 & 4*8 \end{bmatrix} \textit{(2.2.11)}$$

$$min(\boldsymbol{A}, 0) = \begin{bmatrix} min(\mathbf{1}, 0) & min(\mathbf{2}, 0) \\ min(\mathbf{3}, 0) & min(\mathbf{4}, 0) \end{bmatrix} \qquad \textit{(2.2.12)}$$

2.2.2 Linear and Quadratic Equations

As mentioned earlier, a major application of matrices is to represent *linear transformations*, that is, generalizations of linear functions such as $f(x) = 25x + 3$. Consider an equation

$$25\boldsymbol{x} + 15\boldsymbol{y} = 65 . \qquad \textit{(2.2.13)}$$

Reordering and representing coefficients as elements of a matrix, one can rewrite an equation as (2.2.14)

$$[25 \quad 15] \cdot \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} = 65\,. \qquad (2.2.14)$$

On top of that, matrix notation allows to easily represent *systems* of linear equations. As an example, systems of linear equations

$$\begin{aligned} 25\boldsymbol{x} + 15\boldsymbol{y} &= 65 \\ 15\boldsymbol{x} - 30\boldsymbol{y} &= 0 \end{aligned} \qquad (2.2.15)$$

could easily be expressed as

$$\begin{bmatrix} 25 & 15 \\ 15 & 30 \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} = \begin{bmatrix} 65 \\ 0 \end{bmatrix}. \qquad (2.2.16)$$

The nature of such a representation allows to easily extend to higher dimensions. For example, the system of a kind

$$\begin{aligned} a_1 x + b_1 y + c_1 y &= d_1 \\ a_2 x + b_2 y + c_2 y &= d_2 \\ a_3 x + b_3 y + c_3 y &= d_3 \end{aligned} \qquad (2.2.17)$$

could be efficiently rewritten in a matrix form as

$$\begin{bmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \\ a_3 & b_3 & c_3 \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \\ \boldsymbol{z} \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ d_3 \end{bmatrix}. \qquad (2.2.18)$$

Matrix format enables usage of computing power and various specialized methods to solve such systems of equations quickly. Moreover, the former scheme can be modified to express quadratic equations as well.

As an example, consider an expression $1x^2 + 2 \cdot 2xy + 3y^2$. The quadratic form of such an expression is

$$1\boldsymbol{x}^2 + 2 \cdot 2\boldsymbol{xy} + 3\boldsymbol{y}^2 = [\boldsymbol{x} \quad \boldsymbol{y}] \begin{bmatrix} 1 & 2 \\ 2 & 3 \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix}. \qquad (2.2.19)$$

2.2.3 Optimization Routines

Here a brief introduction to conic optimization is provided and software solutions used in practice are discussed.

Conic optimization is a subfield of convex optimization that studies a class of structured convex optimization problems called conic. Without delving into too much details, conic optimization is a field of mathematics applicable to solving portfolio optimization problems: the functions of interest usually satisfy the criteria of convexity.

To be precise, one is usually interested in convex optimization problem of the form

$$\textit{Minimize} \qquad (1/2)\, x^T P x + q^T x \qquad (34)$$

$$\textit{subject to} \qquad Gx \preceq h\,, \qquad (i)$$

$$Ax = b\,, \qquad (ii)$$

where the first part represents the equation to minimize, and the second part represents (i) inequality constraints and (ii) equality constraints. There is nothing special about vector $\boldsymbol{q}$ or matrices $\boldsymbol{P}$, $\boldsymbol{G}$ or $\boldsymbol{A}$: they contain coefficients of the variables of interest.

In practice, the expression studied can be solved using various free (CVXOPT, SciPy, YALMIP) or commercial (Microsoft Excel, MATLAB, R) optimization packages. The CVXOPT package used for the purposes of this work will be covered later in *Programming Concepts* section.

However, in-depth description of specific methods and the math behind them does not fall under the scope of the paper. For detailed overview of optimization routines and cone programming specifically the reader is referred to various papers on mathematical optimization.

2.2.4 Quantopian and Backtesting

Quantopian is a crowd-sourced quantitative investment firm. It provides data, a research environment and a development platform to algorithm authors. Quantopian allows to create, share and test trading algorithms using pre-programmed tools and market data supplied (Quantopian 2017).

The platform provides minute-level price and fundamental data of all US stocks from January 2002 for backtesting. The bar data consists of the high, low, open, close, and volume for each minute that a stock is traded. The price data includes all companies that were traded, including companies that have subsequently gone out of business.

As noted earlier, Quantopian comes in handy when one aims to backtest a certain trading algorithm. Although trading has little to do with investing, the tools developed for traders are convenient for evaluation and testing of investment models.

The advantages of using Quantopian are pretty straightforward. Most importantly, Quantopian supplies specialized trading environment built upon Zipline trading library (Quantopian Inc. 2017b) free-of-charge. Thus, there is no need to purchase or code one. The environment is flexible, highly customizable and comes with a variety of helpful functions.

Moreover, accessing data is easy – Quantopian contains stock records which are adjusted for splits, mergers and dividends as of the simulation date. There is no need to manually clean, configure, setup or store data on a local machine.

Third, the performance measurements and detailed summary of an algorithm efficiency allow for thorough research and analysis of an investment strategy. An important and helpful detail is that the algorithms can easily be shared and replicated should anyone be interested in implementations.

There is a significant academic population using Quantopian today and there are certain reasons for that:

- Quantopian is backed by Zipline, and open-source project which has been reviewed by dozens of authors (Quantopian Inc. 2017b)
- Quantopian is handy for presenting replicable research. Once an account is set, the user can share his/her code, and anyone can copy that code and verify the results
- Quantopian has been already cited in a handful of academic papers
- Quantopian is implemented in Python, which enables the user to leverage the mentioned advantages of the language
- Zipline permits to use an event-based simulation that significantly reduces the risk of look-ahead bias
- Quantopian is used in several universities as a teaching tool (Quantopian Inc. 2017a).

(Dunn 2015.)

2.2.5 Programming Languages and Concepts

Python. Python is a high-level general-purpose programming language that can be applied to many different classes of problems. It is an interpreted, interactive, object-oriented programming language which incorporates modules, exceptions, dynamic typing, very high level dynamic data types, and classes.

Python combines a fair amount of computational power with very clear syntax. It has interfaces to many system calls and libraries, as well as to various window systems, and is extensible in C or C++. It is also usable as an extension language for applications that need a programmable interface. (Python 2017b.)

Advantages. Two very important facts make Python the optimal choice for our work: simplicity and the availability of numerical processing libraries. As stated earlier, the syntax is very brief and intuitive which makes for quick prototyping. Even complicated systems can be assembled and brought on-line relatively fast. Moreover, there are various scientific and financial packages already available (Zipline, NumPy, pandas) – hence no need to code them manually. (Python 2017a.)

Disadvantages. There is an ongoing debate on whether Python is fast. On the one hand, it is interpreted language which makes it somewhat slow compared to compiled counterparts like C++ or C#. On the other hand, a lot of standard library methods in Python are highly optimized; most of the data containers are implemented in C, which makes operations very fast and efficient.

The other limitation is lack of true multithreading - Python is not very suitable language for such a task. Even though there are standard libraries that imitate running multiple threads, the inherent nature of Python (global interpreter lock) does not allow for running multiple processes at the same time.

Pandas package. *pandas* is an open-source Python package which provides fast, flexible, and expressive data structures designed to make working with various data both easy and intuitive. It supplies various specialized methods and data structures fundamental for doing practical data analysis in Python. (Pandas 2017a.)

pandas is well suited for many different kinds of data:

- Tabular data with heterogeneously-typed columns, as in an SQL table or Excel spreadsheet
- Ordered and unordered (not necessarily fixed-frequency) time series data
- Arbitrary matrix data (homogeneously typed or heterogeneous) with row and column labels
- Any other form of observational / statistical data sets

The two primary data structures of *pandas*, Series (1-dimensional) and DataFrame (2-dimensional), handle the vast majority of typical use cases in finance, statistics, social science, and many areas of engineering. *pandas* module is built on top of NumPy and is intended to integrate well within a scientific computing environment with many other 3rd party libraries. (Pandas 2017a.)

Among the many, there are some things useful to us which *pandas* does well:

- Easy handling of missing data (represented as NaN) in floating point as well as non-floating-point data
- Automatic and explicit data alignment: objects can be explicitly aligned to a set of labels, or the user can simply ignore the labels and let Series, DataFrame, etc. automatically align the data in computations
- Make it easy to convert ragged, differently-indexed data in other Python and NumPy data structures into DataFrame objects
- Intelligent label-based slicing, fancy indexing, and subsetting of large data sets
- Robust IO tools for loading data from flat files (CSV and delimited), Excel files and 3rd party databases
- Time series-specific functionality: date range generation and frequency conversion, moving window statistics, moving window linear regressions, date shifting and lagging, etc.

Many of these principles are there to address the shortcomings frequently experienced while using other languages / scientific research environments. As is the case, the introduced procedure shall consist of multiple stages: getting the data, munging and cleaning it, transforming some attributes and processing the results, then organizing the output of the task into a form suitable for plotting or tabular display. *pandas* is a convenient tool for all of these tasks. (Pandas 2017b.)

Some other favourable features:

- *pandas* is fast. Many of the low-level algorithmic bits have been extensively tweaked in Cython code
- *pandas* is a dependency of statsmodels, making it an important part of the statistical computing ecosystem in Python
- *pandas* has been used extensively in production in financial applications

Numpy package. NumPy is the fundamental package for scientific computing with Python. It contains among other things:

- a powerful N-dimensional array object
- sophisticated (broadcasting) functions
- tools for integrating C/C++ and Fortran code
- useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, *NumPy* can also be used as an efficient multi-dimensional container of generic data.

Cvxopt package. CVXOPT is a free software package for convex optimization based on the Python programming language. Its main purpose is to make the development of software for convex optimization applications straightforward and easy. The other good thing about the package is relatively simple syntax and the abundance of practical examples with code available. (CVXOPT 2017.)

3 THE MODEL

3.1 Overview

In this chapter, the actual process of getting the information, transforming it and finding optimal portfolio weights is illustrated and inspected in detail. First, a high-level overview of the developed approach is depicted. As the reader acquires general understanding, nitty-gritty details are presented and a closer look is taken at the little intricacies of the framework.

First step in the procedure is getting and processing financial data. In theory, the data could come in various different forms (Excel tables, csv. files, SQL query response, etc.). The exact format does not really matter so long as the records contain ‘close’ or ‘adj. close’ fields with respective dates, which can be extracted and processed accordingly. The implemented framework of interest assumes that the data comes in *pandas’* DataFrame format (with rows as time observations and columns as securities, see Picture 1) but this can be easily tailored to the concrete implementation case.

If the data comes from different sources, it must be joined on dates to avoid the overlaps and incorrect alignment. Resulting structure must be cleared of non-existing values, which might be the side effect of joining on dates when one security either was not traded or was withdrawn from the trade all together.

Picture 1: *pandas* DataFrame

```
In [6]: pd.DataFrame(np.arange(10*10).reshape((10,10)),index=index)
Out[6]:
             0   1   2   3   4   5
2001-01-01   0   1   2   3   4   5 ...
2001-01-02  10  11  12  13  14  15 ...
2001-01-03  20  21  22  23  24  25 ...
2001-01-04  30  31  32  33  34  35 ...
2001-01-05  40  41  42  43  44  45 ...
2001-01-06  50  51  52  53  54  55 ...
           ... ... ... ... ... ...

[10 rows x 10 columns]
```

What to do with missing values is usually up to a user. There are numerous techniques which range from dropping the observations to forward filling the gaps. The reader, should he or she implement the described procedure, is suggested to exercise caution and adhere to a particular situation and requirements of his / her own case.

After the records are aligned and missing values are dealt with, the data is checked and processed according to the format needed for following estimation and optimization procedures. The rolling returns are necessary for further procedures; they can be calculated in different ways. As the *Quantopian* upper period limit is daily frequency, the rolling daily returns are calculated, checked for missing values and then fed into the optimization module. The prices could also be recorded on a monthly or yearly basis. If that is the case, the format of the prices should either be propagated through the framework or changed at the very beginning to meet the requirements of function discussed further.

Our approach is to compute return distributions and semi-covariance matrices based on daily data and then adjust the result to reflect yearly values. After the values are summarized in a *DataFrame* with dates as index, securities as columns and ‘adj. close’ or ‘close’ prices as values inside the structure.

The next step is invocation of the optimization subroutine. It consists of three small modules: the function computing of the semi-covariance matrix, the function adjusting the expected returns vector and the routine minimizing second-order cone expression. Keep in mind that the matrix evaluation and shrinking procedures do not conflict with each other and therefore their order is of no significance.

The semi-covariance computation method requires a series of asset returns and a benchmark. The benchmark can be chosen according to the particular investor’s needs and beliefs. However, Estrada (2007) proposes to use minimum-variance portfolio expected return value as a benchmark. The mean return of S&P 500 stock universe is utilized.

The Bayes-Stein procedure needs mentioned data structure as well. Once acquired, the simple averages, inverse of covariance matrix and common value are computed and vector of maximum-likelihood expected returns is calibrated.

The resulting semi-covariance matrix and vector of adjusted expected returns is then supplied to the optimization function imported from the CVXOPT optimization package. The resulting vector of weights is optimal and then passed elsewhere. At this point the specifics of the ordering procedure determine the use of the mentioned information.

In our case, the Quantopian built-in function orders the securities and retains their values in a shared data structure to retain its state throughout the lifetime of the simulation. After the simulation is done, the Quantopian automatically computes the number of common performance measures and outputs a statistic on the trading algorithm tested.

3.2 Implementation Details

The following section contains informal high-level description of the operating principles of the presented model. See the Appendix 1 (1-6) for the actual Python implementation.

The specifications are expressed using *pseudocode*. Check the Appendix 2(1) to observe pseudocode conventions used.

3.2.1 Semi-Covariance Matrix Estimation

The former procedure heavily relies on language capabilities and libraries for efficient vector and matrix computations. The curious reader willing to test the model and experiment with the code is strongly suggested to make use of numerical computing packages of his/her language of preference.

The function itself is pretty simple.

```
FUNCTION Semi-Covariance
    Input: A matrix of returns M, a benchmark b
    Output: The semi-covariance matrix out

    FOR each value in a matrix M
        DECREMENT the value by the benchmark b amount
        SET the value to be the result of min(value, 0)
    ENDFOR

    SET new_M to the transpose of M

    COMPUTE out as linear product of two matrices new_M and M
        so that the value at the position out[0][0] equals to the dot
product of
        row vector new_M[0][:] and column vector M[:][0]

    RETURN out
```

The execution begins with a matrix update. First, the interpreter iterates over all values in a given matrix. During the process of iteration, each value is decremented by the benchmark amount supplied at the beginning of a procedure. Then the reduced value is compared to zero. If the resulting value is more than zero, such a value is replaced with zero; otherwise, if the value is negative, nothing is done.

Next, the transpose of a resulting matrix is acquired. When completed, this allows to get the semi-covariance matrix using the properties of matrix multiplication.

The reader might find another implementation which turn out to be faster. In fact, he is encouraged to do so as high efficiency of computations was not our top priority while writing the thesis. In fact, various languages might be better suited for computing.

3.2.2 Bayes-Stein Estimation Procedure

The following implementation relies on numerical processing libraries as well as the former counterparts. As it includes covariance matrix estimation and inversion, which could be pretty expensive in terms of time and resources depending on the size of the dataset, the reader is once again advised to use the tools available in his / her language of choice.

```
FUNCTION Bayes-Stein:
    Input: A matrix of returns M with (t_periods, n_assets) dimensions
    Output: The vector of shrunk estimates of the expected returns out

    COMPUTE the vector means_sample as means along the rows of the
matrix M
        such that means_sample[0] equals to the mean of series M[:][0]
        and means_sample is m-dimensional

    SET ones to be m-dimensional vector of ones
    COMPUTE the E as matrix of pairwise covariances of columns of M

    FOR each value in a matrix E
        MULTIPLY the value by the (t_periods - 1)/(t_periods - n_assets
- 2)
    ENDFOR

    SET the I to be the inverse of matrix E

    COMPUTE the scalar mean_grand as the result of a division, where
        the numerator is the linear product of transposed ones,
            matrix I and vector means_sample,
        and the denominator equals the linear product of transposed ones,
            matrix I and ones

    COMPUTE the vector diff as means_sample - mean_grand

    COMPUTE the scalar denom to be the result of
        (linear product of transposed diff, matrix I and diff)*2 + n +
2

    COMPUTE the scalar w to be the result of
        (n + 2) / denom
    COMPUTE the out as
        (1 - w) * means_sample + w * mean_grand

    RETURN out
```

The procedure of interest is somewhat involved. First, the vector of sample means is computed. The idea is to get the mean return for each asset in a matrix. The described implementation assumes that the rows in the matrix represent points in time, while columns represent separate securities. Therefore, the column means of interest are computed along the rows of the matrix and put in a vector.

Second, the n_assets-dimensional vector of ones is instantiated to assist in future calculations. After that the algorithm computes a new matrix of pairwise covariance of assets so that the entry at $[\boldsymbol{a}][\boldsymbol{b}]$ of the new covariance matrix is a covariance between column vector of returns of asset $\boldsymbol{a}$ at $[:][\boldsymbol{a}]$ and the asset $\boldsymbol{b}$ at $[:][\boldsymbol{b}]$.

After the covariance matrix is adjusted as in Zellner & Chetty (1965) and inversed, the grand mean is computed as in (21). Then, after the differences between the sample means and grand mean are found and put into a vector, the denominator value is computed as in (2123). Further, the weighting coefficient is computed and applied as in (19) to get the vector of adjusted means.

3.2.3 Optimization Procedure

The concrete implementation of the optimization procedure highly depends on a particular software package or module one decides to use. There are pros and cons and modules vary to a large degree, but in order to give at least the feel for how everything will happen, please see the code snippet below.

```
    FUNCTION Find_optimal_portfolio
    Input: A matrix of returns M, a benchmark b, investor's tolerance
tol
    Output: The vector of optimal weights out

    IMPORT function OPTIMIZE from external library
    IMPORT function Semi-Covariance
    Import function Bayes-Stein

    COMPUTE matrix P as the result of Semi-Covariance( M, b)
    COMPUTE vector q as the result of Bayes-Stein( M)

    SET UP the procedure for OPTIMIZE function:
        initialize the equality and inequality constraints as
        G, h, A, b accordingly

    COMPUTE out as a result of OPTIMIZE( P, tol*q, G, h, A, b)

    RETURN out
```

First the user should import the optimization function of his/her choice; the only requirement is that the procedure is suitable for convex optimization. Semi-covariance and Bayes-Stein procedures are imported as well. Then, the semi-covariance matrix $\boldsymbol{P}$ and adjusted vector of returns $\boldsymbol{q}$ are computed.

Usually the optimizers need the equality and inequality constraints – hence setting them in appropriate way. The **out** is then the result of optimization function called with $\boldsymbol{P}$, $\boldsymbol{q}$ and constraints respectively.

4 EXPERIMENTS AND RESULTS

In order to test the efficacy of the suggested method the authors employed a backtesting procedure. The purpose of the backtest is to check the performance of the proposed theoretical prototype on relevant historical data. When done correctly, the results would be a good indicator of whether to utilize an investment strategy or not. Exceptional attention was given to the choice of time periods and sample sizes to ensure the statistical significance and robustness. Although backtesting is not the only or even best way to guarantee the viability of the model in and of itself, it is a good starting point and a powerful tool at our disposal.

4.1 Setup

The tests were conducted for varying risk tolerance levels and different semivariance function parameters using historical data of the stocks constituting the DJIA 30 index (Appendix 3). For simple minimum-variance portfolio the expected returns are calculated as simple means of securities return series, while for the model of interest they are adjusted as in (19). There are three main sets of experiments, each run using the data of specific time period. Different time periods were employed with an aim to test the framework in varying market environments - *bullish* (Jan-Dec/2014), *bearish* (Jan-Dec/2008) and *stagnant* (Jan-Dec/2015) respectively.

In order to inspect the performance of the suggested framework in a bearish market (when the prices are falling and the market is spiralling down in general) the year 2008 was chosen; during that time period, the S&P 500 lost almost 35% of its value and fell from 1378.60 to 902.99 points (NYSEArca 2017). The Jan-Dec/2008 period's universe contains all but 2 securities (Visa Inc. and The Travelers Companies Inc.) as the data on companies is not available for analysis and backtesting, hence no way to train the model before that date. The Jan-Dec/2014 was chosen to represent bullish market conditions (13.5% S&P gain over the year) and Jan-Dec/2015 to represent stagnant market respectively (0.01% S&P 500 yearly gain).

S&P 500 is assumed to be a reasonable approximation of a U.S. stock market and thus treat SPDR S&P 500 Trust ETF's as a proxy for the market's performance. Essentially, same fund's statistics was used to gauge the efficiency of MV- and MS-based portfolios. Risk-free rate level is accepted at 1.09% - current US 12-Month Treasury Bill yield (Bloomberg L.P. 2017). As for the model parameters, preliminary tests were conducted with various cut-off values for semi-covariance matrix benchmark; the results presented here contain only those with 6.9% threshold as the difference between weights resulting from different cut-off values and thus different optimal weight allocations are extremely small. Consequently, the cut-off value used in computations was set to be 6.9%, the YTD return of SPDR S&P 500 Trust ETF (NYSEArca 2017).

Table 4.1.1 contains brief overview of an experimental setup; the results are summarized in tables 4.2.1, 4.2.2 and 4.2.3.

Table 4.1.1: The experimental setup

Testing periods	**Bearish** Jan-Dec/2008, **bullish** Jan-Dec/2014 and **stagnant** Jan-Dec/2015
Market portfolio	SPDR S&P 500 Trust ETF
Risk-free rate	1.09% (US 12-Month Treasury Yield)
Reference portfolio	Zero-risk optimized MV portfolio
Model parameters for Mean-Semivariance with Bayes-Stein estimators	
Risk tolerance values	0.0, 0.25, 0.75, 1.0
Benchmark for semivariance matrix	6.9% (mean return of SPDR S&P 500 Trust ETF)

4.2 Results

Bullish market. Table 4.2.1 presents the results over the Jan-Dec/2014 testing period. Consider the benchmark S&P 500 first. Passive strategy of buying and holding the S&P 500 ETF fund would generate a *mean return* of 13.5% with a *standard deviation* of 11.3% and *Sharpe ratio* at acceptable value of 1.09.

Looking at the sets of variously parametrized MV and MS with shrinkage portfolios, notice that the minimum-risk MV turned out somewhat lacklustre. Although with decent *expected return* of 15.8% and 10.9% *expected variance*, the results in general are inferior to all the alternatives.

On the contrary, the MS w. Bayes-Stein framework achieved consistently better scores across the board. No matter the risk tolerance parameter, it fares efficiently through the year of 2014 with *returns* being 11.7%, 24.3% and 30.2% for risk tolerance parameters 0.0, 0.25 and 0.75 respectively. With moderately high betas and positive alphas, the strategy ranks fairly high according to Sortino ratio, which is expected and welcomed.

Table 4.2.1: Results for simulation in a bullish market; testing period Jan - Dec/2014

	Risk tolerance levels of MS w. B-S.				MV	SPY
	0.0	0.25	0.75	1.0*		
Return, %	11.7	24.3	30.2	-	9.6	13.5
Volatility	0.10	0.13	0.17	-	0.09	0.113
Alpha	0.03	0.12	0.17	-	0.01	~0.0
Beta	0.66	0.83	0.82	-	0.60	~1.0
Sharpe	1.15	1.76	1.67	-	1.05	1.09
Treynor	0.16	0.27	0.35	-	0.14	
Jensen’s alpha	0.06	0.18	0.24	-	0.05	
Sortino	1.72	2.72	2.62	-	1.57	
Max DD, %	-7.4	-8.4	-8.6	-	-6.3	

The *Maximum Drawdown* of a strategy is on a weaker side - even for aversion parameter set at 0.0 the drawdown is -7.4%, which is higher than the MV counterpart. Though the loss does not exceed 10% yearly or even monthly.

Bearish market. Portfolios' dynamics through the year 2008 is shown in Table 2.3. Again, take a look at a market performance first. Those with investments in SPY would lose 36.9% of their wealth with volatility being 39.6%. For the reference, market's Sharpe being -0.94.

Observing the optimized models, note that this time the conservative approach of MV works well: the -16.7% realized return is almost half the market's one with volatility at 28%. As for the new framework, the results are contrasting. The run parametrized by tolerance of 0.0 is a clear winner among the alternatives beating others in every metric listed. On the other hand, for tolerance values differing from 0.0 the performance degenerates at a quick pace. With losses around -50%, high volatilities and almost 60% drawdowns the suggested strategy fails completely.

A curious remark is that at risk tolerance levels of 0.75 and 1.0 MS allocates 100% of the capital to the Apple Inc. security. Even though the projected

Table 4.2.2: Results for simulation in a bearish market; testing period Jan - Dec/2008

	Risk tolerance levels of MS w. B-S.				MV	SPY
	0.0	0.25	0.75	1.0*		
Return, %	-15.1	-52.8	-57.3	-	-16.7	-36.9
Volatility	0.27	0.47	0.58	-	0.28	0.396
Alpha	0.09	-0.28	-0.31	-	0.10	~0.0
Beta	0.57	0.93	0.96	-	0.63	~1
Sharpe	-0.46	-1.35	-1.18	-	-0.52	-0.94
Treynor	-0.28	-0.57	-0.60	-	-0.28	
Jensen's alpha	-0.19	-0.59	-0.63	-	-0.21	
Sortino	-0.69	-1.80	-1.56	-	-0.76	
Max DD, %	-24.1	-58	-58.7	-	-25.4	

returns of Apple Inc. were substantially shrunk, it seems that, given the risk-return trade-offs and semi-covariance matrix, the most optimal allocation is the one above, at least theoretically.

Stagnant market. Table 2.2 shows the performance metrics over the Jan-Dec/2015 time period. Yearly realized market returns are at 1.3% with 15% standard deviation.

On average, both shrunk MS and MV showed better performance than the global benchmark S&P 500. With the exception of strategy parametrized by 0.0 parameter, the performance of the proposed prototype is remarkable: 23.1%, 25.8% and 27.2% for 0.25, 0.75 and 1.0 risk aversion parameters. Alphas and Treynor are moderate with Sortino being high at around 2.0; somewhat high betas at around 1.0 are the only drawback.

The minimum-risk (parameter 0.0) shrunk MS system is the worst among the counterparts with realized loss at -0.9% and high volatility of 27%. A fact worth noting is that the weight allocations of MV and 0.0 tolerance MS differ drastically.

Table 4.2.3: Results for simulation in a stagnant market; testing period Jan - Dec/2015

	Risk tolerance levels				MV	SPY
	0.0	0.25	0.75	1.0		
Return, %	-0.9	23.1	25.8	27.2	0.8	0.013
Volatility	0.27	0.18	0.19	0.19	0.14	0.15
Alpha	-0.02	0.20	0.22	0.23	0.0	~0.0
Beta	0.80	0.95	0.97	0.98	0.83	~1.0
Sharpe	-0.93	1.23	1.32	1.35	0.12	0.01
Treynor	-1.13	0.23	0.25	0.26	0.08	
Jensen’s alpha	-0.95	0.16	0.19	0.20	0.02	
Sortino	0.00	1.86	2.05	2.12	0.18	
Max DD, %	-14.4	-11.5	-9.7	-9.6	-12.7	

5 SUMMARY AND CONCLUDING REMARKS

5.1 Summary

In this paper, the mean-semivariance optimization scheme that employs Bayes-Stein estimators is proposed, implemented and studied. The model is examined through empirical experiments with 30 U.S. stocks representing DJIA and is compared to zero-risk mean-variance scheme and S&P 500 benchmark under variant parameters and time periods.

The results in general support the superiority of the MS framework employing Bayes-Stein estimators while being somehow mixed: based on the outcomes of backtests, the prototype fared substantially better than the Markowitz' MV and SPY ETF in the bullish and stagnant market settings, but was considerably worse in the bearish market case for the majority of risk tolerance parameters. Most probably such an outcome originates from the improved expected returns and downside risk measure introduced into the model. In addition, the experiments suggest that (1) the performance of the MS does not vary too greatly with the choice of benchmark and (2) the allocations are less sensitive for the risk tolerance values above the certain threshold.

The results generally support the findings of the previous studies - the mean-semivariance model with shrinkage does indeed produce superior results. However, it is fairly difficult to pinpoint the concrete feature which gives us the edge over the conventional methods; it is unclear whether downside framework or shrinkage bring/s is the source of the additional alpha.

Although promising, the performance of the model needs to be put under additional tests and research. Various testing conditions as well as parameters and testing methods must be applied before using the proposed method in practice. In particular, the recommendation would be to stress-test the approach with Monte Carlo simulation employing stochastic asset model.

5.2 Discussion

Experiments conducted provide useful information for an efficient MS portfolio scheme. The realized returns were greater when using proposed system. The reasons for this are, firstly, improved accuracy of projections of expected returns and, secondly, different risk estimate introduced with semivariance. By introducing Bayes-Stein procedure the estimation error stemming from the randomness and uncertainty about future returns are reduced. With semivariance as risk measure relative riskiness of the assets is changed, thus expanding the pool of available optimal investment options. Indeed, in general, the framework of interest shows an improved performance over the benchmarks in certain market conditions, namely, the bearish and bullish environments. The performance measures strongly suggest the effectiveness of the MS with Bayes-Stein estimators.

However, the situation is somewhat different in a declining market state: except for the minimum-risk case the realized returns and volatilities are twice as bad as the S&P 500. The key to understanding such a performance lies in the characteristics of the positions opened. With less aversion, the model took the riskier approach. The assets purchased could have been affected by market crash the most; high betas are the evidence.

An interesting note is that during some periods the system assigns 100% weight to a particular security or a couple at most. On the one hand, by doing this the best risk-return ratio can be achieved. On the other hand, the concept of diversification does not apply to such a portfolio composition, hence the susceptibility to the company-specific risk.

Further, MS with shrinkage is almost insensitive to changes in the semi-covariance cut-off value above zero: the results between 0.0%, 1.09% and 6.97% do vary, but the weight differences in final allocations are insignificant. As the testing capital was set to be $100.000, assets whose optimal weights fell below certain threshold could not be purchased.

5.3 Constraints and Future Work

With lack of knowledge and expertise being the case, certain limitations are accepted and a number of constraints imposed on the research.

Constraints. Some common research assumptions used in finance were made. While testing the prototype short-selling scenarios are completely excluded; short-selling brings additional risk and is a fairly delicate tool to use. Further, the absence of friction costs is accepted as they partly depend on choice of a broker. The assets of interest were deemed to be always available for purchase. In our tests, the performance of S&P 500 was accepted as a proxy for the U.S. stock market; the U.S. 12-Month Treasury Bill rate was taken as a risk-free rate.

Limitations. The system itself is based on a single-period investment strategy without rebalancing. Only one testing scheme was used to assess the efficiency of the suggested approach. Backtesting was conducted in U.S. stock market environment, hence lack of international diversification. Stock universe selection might be debatable: disturbingly often both MV and MS chose a small number of stocks to assign weights to, leading to the lack of diversification. Moreover, the presence of certain performance outliers might have influenced the weight allocations; for example, at some point AAPL was assigned a 1.00 weight by both MV and MS models. Training and testing periods were fixed to be rather small in order to reflect latest information about the asset; although, this could have led to insufficient amount of data to base the analysis on.

Future work. This chapter contains some possible ways to proceed with the investment strategy of interest. There are numerous paths to take and the majority of them lies in a realm of further testing. Many possible tests, adaptations and experiments have been left for the future due to lack of time and knowledge. Future work concerns deeper analysis of the capabilities, new test runs or simple curiosity.

The following is just a handful of ideas to apply when pursuing the suggested approach:

- Test using different benchmark values in MS matrix computation (risk-free asset, value of zero, etc.)
- Choose richer range of tolerance parameters for MV and MS
- Test with Bayes-Stein and without Bayes-Stein adjustment
- Pick various ranges of data to train and test on
- Enrich the universe of assets: check different industry sectors, various countries (not only US), sort by liquidity; throw more stocks in a mix
- Employ more testing schemes to use (Grauer & Hakansson 1995, 55) - Monte Carlo Simulation

BIBLIOGRAPHY

Barry, C. B. 1974. Portfolio analysis under uncertain means, variances, and covariances. The Journal of Finance, 292, 515-522.

Black, F., & Litterman, R. B. 1991. Asset allocation: combining investor views with market equilibrium. The Journal of Fixed Income, 12, 7-18.

Bloomberg L.P. 2017. United States Rates & Bonds. Accessed 17 May 2017. https://www.bloomberg.com/markets/rates-bonds/government-bonds/us.

Blyth, C. R. 1951. On minimax statistical decision procedures and their admissibility. *The Annals of Mathematical Statistics*, 22-42.

Broadie, M. 1993. Computing efficient frontiers using estimated parameters. Annals of Operations Research, 451, 21-58.

Brown, S. J. 1976. Optimal portfolio choice under uncertainty: A Bayesian approach Doctoral dissertation, University of Chicago, Graduate School of Business.

Bryman, A., Lewis-Beck, M. S., & Liao, T. F. 2004. The SAGE encyclopaedia of social science research methods. Sage Pub.

Cavadini, F., Sbuelz, A., & Trojani, F. 2001. A Simplified Way of Incorporating Model Risk, Estimation Risk and Robustness in Mean Variance Portfolio Management.

Ceria, S., & Stubbs, R. A. 2016. Incorporating estimation errors into portfolio selection: Robust portfolio construction. In Asset Management pp. 270-294. Springer International Publishing.

Cerin, P., & Dobers, P. 2001. What does the performance of the Dow Jones Sustainability Group Index tell us? Eco-Management and Auditing, 83, 123-133.

Cherney D., Denton T., Thomas R. & Waldron A. 2013. Linear Algebra. First Edition. Davis California. Accessed 9.5.2017. https://www.math.ucdavis.edu/~linear/linear-guest.pdf.

Chopra, V. K., & Ziemba, W. T. 2011. The effect of errors in means, variances, and covariances on optimal portfolio choice. The Kelly Capital Growth Investment Criterion: Theory and Practice, 3, 249.

CVXOPT. 2017. Home - CVXOPT. Accessed 9.5.2017. http://cvxopt.org/.

Dickinson, J. P. 1974. The reliability of estimation procedures in portfolio analysis. Journal of Financial and Quantitative Analysis, 903, 447-462.

Dunn D. 2015. Writing a thesis on pair trading strategies. Should I use Quantopian for research? Accessed 20 May 2017. https://www.quantopian.com/posts/writing-a-thesis-on-pair-trading-strategies-should-i-use-quantopian-for-research.

Efron, B., & Morris, C. 1975. Data analysis using Stein's estimator and its generalizations. Journal of the American Statistical Association, 70350, 311-319.

Efron, B., & Morris, C. N. 1977. Stein's paradox in statistics.

Estrada, J. 2002. Systematic risk in emerging markets: the D-CAPM. Emerging Markets Review, 34, 365-379.

Estrada, J. 2006. Downside risk in practice. Journal of Applied Corporate Finance, 181, 117-125.

Estrada, J. 2007a. Mean-semivariance behaviour: Downside risk and capital asset pricing. International Review of Economics & Finance, 162, 169-185.

Estrada, J. 2007b. Mean-semivariance optimization: A heuristic approach.

Frankfurter, G. M., Phillips, H. E., & Seagle, J. P. 1971. Portfolio selection: the effects of uncertain means, variances, and covariances. Journal of Financial and Quantitative Analysis, 1251-1262.

Grauer, R. R., & Hakansson, N. H. 1995. Stein and CAPM estimators of the means in asset allocation. *International Review of Financial Analysis*, *4*(1), 35-66.

Harlow, W. V. 1991. Asset allocation in a downside-risk framework. Financial Analysts Journal, 28-40.

Hicks, J. R. 1962. Liquidity. The Economic Journal, 72288, 787-802.

Hodges, J. L., & Lehmann, E. L. 1951. Some applications of the Cramer-Rao inequality. In *Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability*. The Regents of the University of California.

Horst, J., De Roon, F., & Werker, B. J. 2002. Incorporating estimation risk in portfolio choice.

James, W., & Stein, C. 1961, June. Estimation with quadratic loss. In Proceedings of the fourth Berkeley symposium on mathematical statistics and probability Vol. 1, No. 1961, pp. 361-379.

Jobson, J. D., & Korkie, B. 1980. Estimation for Markowitz efficient portfolios. Journal of the American Statistical Association, 75371, 544-554.

Jobson, J. D., Korkie, B., & Ratti, V. 1979. Improved estimation for Markowitz portfolios using James-Stein type estimators. In Proceedings of the American Statistical Association, Business and Economics Statistics Section Vol. 41, pp. 279-284. American Statistical Association Washington DC.

Jorion, P. 1986. Bayes-Stein estimation for portfolio analysis. Journal of Financial and Quantitative Analysis, 2103, 279-292.

Klein, R. W., & Bawa, V. S. 1976. The effect of estimation risk on optimal portfolio choice. Journal of Financial Economics, 33, 215-231.

Ledoit, O., & Wolf, M. 2003. Improved estimation of the covariance matrix of stock returns with an application to portfolio selection. Journal of empirical finance, 105, 603-621.

Levin M. 2017. Python Programming Language Advantages & Disadvantages in 2017. Accessed 20 May 2017. http://mikelev.in/2011/01/python-programming-language-advantages/.

Markowitz, H. M. 1968. Portfolio selection: efficient diversification of investments.

Markowitz, H. M. 1991. Foundations of portfolio theory. The Journal of Finance, 462, 469-477.

Michaud, R. O. 1989. The Markowitz optimization enigma: Is optimized optimal? ICFA Continuing Education Series, 19894, 43-54.

Michaud, R. O. 1999. Efficient Asset Management: A Practical Guide to Stock Portfolio Optimization and Asset Allocation. Harvard Business School Press, Boston, MA.

Mueller, P. A., Padmaraj, R. A., & St. John, R. C. 1999. The Dow Jones Industrial Average: Issues of Downward Bias and Increased Volatility. American Journal of Business, 141, 41-52.

NumpPy. 2017. NumPy – NumPy. Accessed 20 May 2017. http://www.numpy.org/.

NYSEArca. 2017. SPDR S&P 500 ETF (SPY) Performance Overview. Accessed 18 May 2017. https://finance.yahoo.com/quote/SPY/performance?p=SPY.

Pandas. 2017a. Package overview - pandas 0.20.1 documentation. Accessed 20 May 2017. http://pandas.pydata.org/pandas-docs/version/0.20/overview.html.

Pandas. 2017b. Pandas: powerful Python data analysis toolkit. Accessed 20 May 2017. http://pandas.pydata.org/pandas-docs/version/0.15.2/index.html.

Platt, H. D., Cai, L., & Platt, M. A. 2014. Is the DJIA Index Biased? The Journal of Index Investing, 44, 43-52.

Pogue, G. A. 1970. An Extension Of The Markowitz Portfolio Selection Model To Include Variable Transactions' costs, Short Sales, Leverage Policies And Taxes. The Journal of Finance, 255, 1005-1027.

Porter, R. B. 1974. Semivariance and stochastic dominance: A comparison. The American Economic Review, 641, 200-204.

Python. 2017a. Python Speed. Accessed 20 May 2017. https://wiki.python.org/moin/PythonSpeed.

Python. 2017b. Welcome to Python. Accessed 20 May 2017. https://www.python.org/.

Quantopian Inc. 2017a. Academia - Quantopian. Accessed 20 May 2017. https://www.quantopian.com/academia.

Quantopian Inc. 2017b. GitHub - quantopian/zipline: Zipline, a Pythonic Algorithmic Trading Library. Accessed 20 May 2017. https://github.com/quantopian/zipline.

Scherer, B. 2002. Portfolio resampling: Review and critique. Financial Analysts Journal, 98-109.

Sharpe, W. F. 1963. A simplified model for portfolio analysis. Management science, 92, 277-293.

Sharpe, W. F. 1964. Capital asset prices: A theory of market equilibrium under conditions of risk. The Journal of Finance, 193, 425-442.

Sortino, F. A., & Price, L. N. 1994. Performance measurement in a downside risk framework. the Journal of Investing, 33, 59-64.

Sortino, F. A., & Van Der Meer, R. 1991. Downside risk. The Journal of Portfolio Management, 174, 27-31.

Statman, M. 1987. How many stocks make a diversified portfolio? Journal of Financial and Quantitative Analysis, 2203, 353-363.

Stein, C. 1956, January. Inadmissibility of the usual estimator for the mean of a multivariate normal distribution. In Proceedings of the Third Berkeley symposium on mathematical statistics and probability Vol. 1, No. 399, pp. 197-206.

Tobin, J. 1958. Liquidity preference as behaviour towards risk. The review of economic studies, 252, 65-86.

Zellner, A., & Chetty, V. K. 1965. Prediction and decision problems in regression models from the Bayesian point of view. Journal of the American Statistical Association, 60310, 608-616.

APPENDICES

Appendix 1: Quantopian implementation

Recall that Quantopian is built on top of Zipline, so some function calls might appear strange and out-of-nowhere; such function are marked with red color and are built-ins. However, they are pretty self-explanatory; treat them as if they were defined somewhere else and imported automatically.

```
"""
Mean-semivcovariance optimization routine (Bachelor's thesis).

Adjust expected return vectors via Jorion's Bayes-Stein technique
(1986), compute Estrada's semi-covariance matrix (2007) and then use
it to find optimal portfolio.
"""

import numpy as np
import cvxopt as opt
from cvxopt import blas, solvers

# to pretty-print the numbers
np.set_printoptions(formatter={'float_kind':'{:0.3f}'.format})

def initialize(context):
    dow_jones = [
        sid(4922),
        sid(679),
        sid(24),
        sid(698),
        sid(1267),
        sid(23112),
        sid(1900),
        sid(4283),
        sid(2119),
        sid(8347),
        sid(3149),
        sid(20088),
        sid(3496),
        sid(3766),
        sid(3951),
        sid(4151),
        sid(25006),
        sid(4707),
        sid(5029),
        sid(5061),
        sid(5328),
        sid(5923),
        sid(5938),
        sid(7792),
        sid(7883),
        sid(21839),
        sid(8229),
        sid(2190)
        ]

    context.securities = dow_jones

    context.counter = 1
```

```
    schedule_function(my_rebalance, date_rules.month_start(),
time_rules.market_open(hours=1))

    # Record tracking variables at the end of each day.
    # uncomment to call recording procedure
    """schedule_function(my_record_vars, date_rules.every_day(),
time_rules.market_close())"""


def my_record_vars(context, data):
    """
    Plot variables at the end of each day.
    """

    record('Portfolio risk (monthly)', context.risk,
           'Portfolio expected return (monthly)',
context.portfolio_return)


def my_rebalance(context, data):
    """ Called once on the first month's trading day, 1 hour after
market is open; constructs the model and orders securities.

    Manually set up the length_years for length of the training period
    and pass freq={'d'/'m'/'y'} to the find_optimal_portfolio training
    function.
    Uncomment the logs for the log output."""

    # parameters
    bench = 0.07  # benchmark for the semicov procedure
    freq = 'y'  # frequency of data we are interested in
    tol = 1.  # risk rolerance
    length_years = 2  # the length of a training period

    if context.counter != 1:
        return 0

    df = data.history(context.securities, 'close', 365*length_years,
'1d')
    #log.info('The dimensions of the dataframe: {} \nNumber of missing
values: {}'.format(df.shape, df.isnull().sum()))
    df.dropna(inplace=True)
    returns = df.pct_change()
    returns.dropna(inplace=True)

    #log.info('The length of the training period: {}
year/s.\nDimensions of the returns vector after dropnas:
{}\n'.format(length_years, returns.shape))

    # get optimal weights, portfolio return and portfolio risk
    optimal_weights, context.portfolio_return, context.risk  =
find_optimal_portfolio(returns, benchmark=bench, freq=freq,
tolerance=tol)
    sharpe = (context.portfolio_return - 0.0275) / context.risk

    #log.info('Portfolio expected return: {} \nRisk: {} \nSharpe: {}
\nWeights:{} \n\n'.format(context.portfolio_return, context.risk,
sharpe,optimal_weights))
    order_securities(context.securities, optimal_weights)

    context.counter += 1
    return 1
```

```
def order_securities(securities, weights):
    open_orders = get_open_orders()
    for stock, target in zip(securities, weights):
        if stock not in open_orders:
            order_target_percent(stock, target)
    return 1



def semicov(matrix, benchmark=0.0):
    """Estimate semi-covariance matrix, given data.

    Semi-covariance indicates the level to which two variables vary
together, given that
    these variables are less than the benchmark. If we look at N-dim
samples, X = [x_1, x_2, ... x_N],
    then the semicov matrix element SC_{ij} is the semi-covariance of
x_i and x_j. The element
    C_{ii} is the semivariance of x_i

    Parameters
    ----------
    matrix : ndarray
        Matrix is a 1-D or 2-D array_like containing multiple
variables and observations.
        Each column represent a variable, and each row a single
observation of all those variables.
        [
        [a1, b1, c1, d1, ...],
        [a2, b2, c2, d2, ...],
        [a3, b3, c3, d3, ...],
        ...
        ]
    benchmark: float64 (default 0.0)
        Benchmark return is an arbitrary value for the semicov matrix
estimation. Return percentage in decimals.

    Returns
    -------
    semicov_matrix : ndarray
        The 2-D semi-covariance matrix of the variables.

    """

    if matrix.ndim > 2:
        raise ValueError("matrix has more than 2 dimensions")

    newM = np.array(matrix, ndmin=2)
    shape = newM.shape
    newM -= benchmark  # compute the difference between asset and
benchmark returns
    zero = np.zeros(shape)
    newM = np.fmin(newM, zero)  #  apply min(observation, 0) to
observations
    t_periods = shape[0]  # get total number of observations
```

```
    newM_T = newM.T
    semicov_matrix = newM_T.dot(newM) # first goes the transposed
matrix!
    semicov_matrix *= 1. / np.float64(t_periods)  # divide to get
expected semicovs

    return semicov_matrix


def estimate_bayes_stein(matrix, adjust=True):
    """Return a vector of Jorion's Bayes-Stein estimates for columns
in matrix.

    Parameters
    ----------
    matrix : array_like
        2-dimensional numpy array of a kind
        [
        [a1, b1, c1, d1, ...],
        [a2, b2, c2, d2, ...],
        [a3, b3, c3, d3, ...],
        ...
        ], where each columns represents a varaible, and row
represents observation

    Returns
    -------
    adjusted_means : ndarray
        1-dimensional np.array of a kind [a_shrunk, b_shrunk,
c_shrunk, d_shrunk, ...].

    """

    t, n = matrix.shape  # (number of observations, number of assets)
tuple
    sample_means = matrix.mean(axis=0)
    #log.info('Non-adj:{}'.format( (sample_means + 1)**251-1))
    ones = np.ones_like(sample_means)

    E = np.cov(matrix, rowvar=False)
    adj_Z_C = (float(t) - 1) / (t - n - 2)
    E *= round(adj_Z_C, 4)  # Zellner & Chetty adjustment of sample
cov matrix

    I = np.linalg.inv(E)
    grand_mean = sample_means.dot(I).dot(ones.T) /
ones.dot(I).dot(ones.T)
    diff = sample_means - grand_mean
    denominator = n + 2 + diff.dot(t*I).dot(diff.T)
    w = round((n + 2) / denominator, 4)
    #log.info('The weighting coefficient in Bayes-Stein shrinkage :
{}'.format(w))
    adjusted_means = (1 - w)*sample_means + w*grand_mean

    return adjusted_means
```

```
def find_optimal_portfolio(df, benchmark=0.0, freq='y',
tolerance=0.0):
    """Given the matrix of returns, a benchmark and a frequency,
compute and return a tuple (ndarray of optimized weights, expected
portfolio return, expected risk).

    Parameters
    ----------
    df: pandas DataFrame
    benchmark: float 64 (default 0.0)
    freq: string (default 'y')
        Specify the adjustment frequency of observations.
        'd' for daily returns and semicovs, 'm' for monthly and 'y'
for yearly.
    tolerance: float64 -- [0, 1] (default 0.0)
        Investor's risk tolerance. Arbitrary value which specifies a
degree to which an investor is risk-averse. 0 for no tolerance
(minimal risk), 1 for full tolerance.

    Returns
    -------
    out: tuple of form (ndarray, ndarray, ndarray)
        Function returns (optimal_weights, portfolio_return,
portfolio_risk) tuple.

    """

    trading_days = {'d': 1,
                    'm': 25,
                    'y': 251}
    benchmark = round((benchmark + 1)**(1./trading_days[freq]) - 1, 6)
# convert yearly benchmark to daily one

    solvers.options['show_progress'] = False
    n = df.shape[1]  # number of assets
    values = df.values

    # semi-covariance matrix estimation
    S = semicov(values, benchmark)  # daily semi-covariance matrix
    S *= trading_days[freq]  # daily / monthly / yearly semicov matrix

    # bayes-stein adjustment
    r_adj = estimate_bayes_stein(values)  # vector of adjusted
expected returns
    r_adj = (r_adj + 1)**trading_days[freq] - 1  # daily / monthly /
yearly expected returns

    P = opt.matrix(S)
    q = opt.matrix(r_adj)
    G = - opt.matrix(np.eye(n))  # left side of inequality constraints
- weights' coefficients
    h = opt.matrix(0.0, (n,1))  # right side - the constraint itself -
all weights >= 0
    A = opt.matrix(1.0, (1,n))  # left side of equality constraints
    b = opt.matrix(1.0)  # right side - sum of weights equal to 1

    """log.info('Passed arguments: \nAdjsuted for daily values
Benchmark = {},
    #\nFreq = {}, \nTolerance = {}\nAdjusted Returns vector: {}\n
    #Optimizing ...\n\n\n'.format(benchmark, freq, tolerance,
r_adj))"""
```

```
    # optimization procedure
    weights = solvers.qp(2*P, tolerance*(-q), G, h, A, b)['x']  # note
the tolerance parameter
    portfolio_return = blas.dot(q, weights)
    risk = np.sqrt(blas.dot(weights, P*weights))
    out = np.array(weights), np.array(portfolio_return),
np.array(risk)

    return out
```

Appendix 2: Pseudocode Conventions

Pseudocode is an artificial and informal language that helps to develop algorithms. It allows to focus on the logic of the algorithm without being distracted by details of language syntax. The rules are reasonably straightforward.

The sequential progression (linear path from the beginning till the end of execution) of an algorithm is indicated by writing one action after another, each on a distinct line, and all of them aligned by the same indent. They are performed in the order (from top to bottom) they are written.

Keyword	Description
COMPUTE	Perform an operation and initialize a result
DECREMENT,MULTIPLY	Perform *in-place* arithmetic on a certain variable
FOR … ENDFOR	Show the beginning and the end of a “counting loop”, allowing repeated execution of the code in the main body
IMPORT	Import external module or a class
Input: … / Output: …	Indicate the parameters passed as function arguments
SET	Initialize the variable in a specified way
SET UP	Initialize required parameters for a function

Appendix 3: DJIA components

3M: MMM
American Express: AXP
Apple: AAPL
Boeing: BA
Caterpillar: CAT
Chevron: CVX
Cisco Systems: CSCO
Coca-Cola: KO
DuPont: DD
ExxonMobil: XOM
General Electric: GE
Goldman Sachs: GS
The Home Depot: HD
IBM: IBM
Intel: INTC
Johnson & Johnson: JNJ
JPMorgan Chase: JPM
McDonald's: MCD
Merck: MRK
Microsoft: MSFT
Nike: NKE
Pfizer: PFE
Procter & Gamble: PG
Travelers: TRV
UnitedHealth Group: UNH
United Technologies: UTX
Verizon: VZ
Visa: V
Wal-Mart: WMT
Walt Disney: DIS