Liberà Università degli Studi Sociali Guido Carli
Department of Economics and Finance
Bachelor in Economics and Business
Major in Finance

A Monte Carlo Simulation applied to Portfolio Management and the Constant Proportion Portfolio Insurance

Supervisor:
Professor Emanuele Tarantino

Candidate:
Dario Faggi
236041

Academic year: 2020/2021

A Lodovico, a Dina, a Gemma e a tutta la mia famiglia

Abstract

The practice of investment management has been transformed in recent years by new computational methods. Statistical Machine Learning tools, novel algorithmic representations of data and increasing computational power have allowed for non-parametric extensions of financial modelling that are characterized by a better out-of-sample forecasting process with respect to Modern Portfolio Theory predictions. The programming language implementation of portfolio optimization as proposed by Markowitz has given a new perspective about the roles played by parameter estimates such as expected return and the co-variance matrix in the asset allocation process. When Portfolio Managers apply quadratic optimization, most of the times they may run into several problems. For instance, the weights of the Optimal Portfolio vary wildly with small changes in the estimates of returns and covariances parameters. Also, the mathematical resolution of the Markowitz optimization problem may lead to portfolios that show little diversification, whose out-of-sample performances are poor, or whose capital allocation is illogical from an economic standpoint. The expected returns estimates are too error-prone; they may work against risk diversification and favour disappointing out-of-sample Sharpe Ratios. Actually, these problems can become so severe that many Portfolio Managers prefer to completely ignore the standard optimization process and replace it with advanced methodologies such as the risk parity allocation strategy. This thesis develops a portfolio optimization approach by using a Monte Carlo simulation and describes how it can be implemented with linear programming. The proposed method is tested in Python with a portfolio composed of 15 random stocks to underline some of the major drawbacks of the Markowitz Analysis and the benefits derived from the alternative procedure in terms of forward-looking results. Lastly, this thesis illustrates the importance of the Constant Proportion Portfolio Insurance as a basic algorithm in re-balancing strategies that are widely adopted in the financial industry. This strategy may help a Portfolio Manager in creating option-like payoff functions by exploiting upside potential while limiting the risk of sever drawdowns.

Contents

Chapter 1

Introduction

1.1 Background

The robustness of optimization approaches is of major concern for Portfolio Managers. Slight changes in the estimation of return and risk parameters may drastically alter the portfolio composition and the portion of capital to be allocated to risky financial assets. The best-know portfolio optimization process was illustrated in Markowitz's article published 1952 in the Journal of Finance [16]. However, the application of his optimization model has showed several drawbacks. In particular, the lack of robustness of Markowitz Analysis with respect to errors in parameter estimates is one of the major pitfalls in its implementation.

Traditionally, estimation errors has been the key challenge in portfolio optimization. This problem is particularly important when it comes to expected return estimates because they are much harder to be obtained with a good degree of accuracy if compared to variance/covariance matrix estimates. Asset managers and investors are more likely to focus on portfolio construction methodologies that do not heavily rely on those parameter estimates. In particular, they tend to avoid expected return estimates and focus on the Global Minimum Variance Portfolio, since it is the only portfolio on the efficient frontier for which no expected return estimates are needed.

1.2 Aim and Purpose

This thesis provides an introduction to the Investment Management science, with a particular emphasis on the use of data science techniques to improve investment decision. We will discuss hands-on implementations of recent applications of techniques to portfolio management decisions, including the design of more robust empirical models, the construction of portfolios with improved diversification benefits through a Monte Carlo Simulation, and the implementation of a particular insurance Portfolio strategy.
Instead of merely explaining these concepts, we will cover a practical analysis of those ideas in the Python language. We will analyze the problem of estimating risk and return parameters for meaningful portfolio decisions and illustrate the benefit of applying a Monte Carlo simulation to the process of portfolio construction and analysis.
Finally, we will develop a primary insurance strategy that represents a core algorithm in investment management and portfolio construction due to its enhanced robustness. In Chapter 3, the strategy will be tested on possible Brownian developments of our 15-stocks portfolio to show how a Portfolio Manager may exploit upside potential of a risky asset by maintaining a downside protection at the same time.

1.3 Methodology

Firstly, the traditional concepts and methods of Modern Portfolio Theory are analyzed. Therefore, relevant assumptions and data manipulations are illustrated to implement a 4-stocks portfolio optimization in Python. Secondly, the methodologies of the Monte Carlo simulation and the properties of the Geometric Brownian Motion model are described. Consequently, an optimization algorithm that executes a Monte Carlo simulation is implemented to find the Maximum Sharpe Ratio and the Global Minimum Variance Portfolio composed by 15 randomly-chosen stocks. Thirdly, different scenarios for the evolution of the Max Sharpe Ratio Portfolio are generated through a Random Walk model. Finally, a comparison is drawn between the statistical performances of a portfolio that is re-balanced once a month according to the Constant Proportion Portfolio Insurance principles and one that is left undisturbed.

1.4 Thesis Structure

The thesis is structured into the following four chapters:

- In Chapter 1, the background of current optimization approaches is briefly intro-

duced, the aim and the purpose of the thesis are presented and the implemented methodology is described.

- In Chapter 2, the fundamental concepts of Modern Portfolio Theory are defined and the result of a portfolio optimization carried out in Python according to Markowitz's principles is analyzed. Additionally, we show how the shape of the Efficient Frontier is dramatically altered when a risk-free asset is introduced.

- In Chapter 3, a Monte Carlo simulation technique is applied to the problem of portfolio optimization. The associated algorithm is introduced in full details and compared to the standard portfolio optimization procedure. The empirical weights of the Maximum Sharpe Ratio and Global Minimum Variance Portfolio are founded and confronted with the theoretical ones.

- In Chapter 4, the expected return and volatility of the Maximum Sharpe Ratio Portfolio founded in Chapter 3 are used to produce ten possible developments of the portfolio over the next 252 trading days according to a Brownian Motion Model. We also analyze the crucial role played by the Constant Proportion Portfolio Insurance strategy in the context of portfolio management by comparing 2 possible forward-looking performances of the same portfolio.

Chapter 2

Overview of Modern Portfolio Theory

Many of the arguments covered in this Chapter date back to a world-wide famous article written in 1952 by Harry Markowitz [16], the father of Portfolio Selection and the 1990 Nobel Prize winner in Economics. This publication led the foundation of a mathematical framework called Modern Portfolio Theory (MPT) that is still widely used among mutual funds, pension plans, banks, insurance companies and many others institutions. Chapter 2 analyzes the relevant characteristic of asset securities and portfolios, implements Markowitz's mathematical model by offering an example of portfolio optimization in Python and ultimately discusses some of the criticisms against Markowitz's criteria for selecting an efficient portfolio.

2.1 Fundamentals

2.1.1 Actual and Expected Returns

The Actual Dollar Return of an asset can be defined as the total dollar value that an investor gains or loses over a certain period of time. It is the sum of the cash received and the change in the value of the asset, in dollars term

$$DollarReturn = C_{t+1} + P_{t+1} - P_t \tag{2.1}$$

where:

- C_{t+1} is any Cash Flow that directly derives from the asset during the period
- P_{t+1} is the price at the end of the period
- P_t is the price at the beginning of the period

In an equity-purchase context, the percentage return of a stock at time t (R_t) is given by the dividend yield plus any capital gain or loss:

$$R_t = \frac{P_{t+1} - P_t}{P_t} + \frac{D_{t+1}}{P_t} \tag{2.2}$$

where $D_{t+1}$is the dividend paid during the period.
On the other hand, an expected return is uncertain by definition. It is based on different types of estimations and may or may not occur in the future. The reason for that is the concept of risk, that is the unpredictability of future returns from an investment [18]. One of the most common unbiased estimator under the Efficient Market Hypothesis is the arithmetic average of the historical returns [14]:

$$E[R] = \frac{1}{N} \sum_{t=1}^{N} R_t \tag{2.3}$$

where R_t is the return at time t and N is the number of trading days.
Given that in Chapter 3 we will develop a Monte Carlo Simulation based on the Geometric Brownian Motion (GBM) Model, a log-normal returns approach seems to be the most appropriate [15]. We define the logarithmic return of a stock at time t as

$$R_t = \log\left(\frac{P_{t+1}}{P_t}\right) \tag{2.4}$$

where P_t is the price of the stock at time t.
As we will mention in Sec. 2.2.1, the use Log Returns also implies great statistical advantage and smoothness in computational procedure.

2.1.2 Portfolio return

Under the MPT assumptions, expected returns of any weighted combination of assets is just the weighted average of the expected returns of those assets:

$$E(R_p) = \sum_{i=1}^{N} w_i E(R_i) \tag{2.5}$$

where

- R_p is the return of the portfolio
- w_i is the weight of asset i in the portfolio
- R_i is the return of asset i

In other words, the portfolio expected return is simply the arithmetic mean of the returns of the individual securities weighted by their portfolio weights.

2.1.3 Volatility as the simplest measure of risk

A general accepted trading principle, the Risk-Return Trade-off, claims that there is usually a positive relationship between the level of risk and the potential return. The higher the level of risk, the greater is the potential return. Volatility quantifies the amount of risk of a certain investment. Under the assumption of normally-distributed level of prices in the stock market, the standard deviation of the returns is the simplest statistical figure that represents volatility. The standard deviation δ_k of the returns of asset k on a risky investment provides a measure of the deviation of the values of k from their mean μ_k [8] and it is defined as follow:

$$\delta_k = \sqrt{\frac{1}{N-1}\sum_{t=1}^{N}(k_t - \mu_k)^2} \tag{2.6}$$

where:

- k_t represents the returns of asset k at time t
- N is the number of returns
- μ_k is the expected return of asset k

(see Eq.2.3) [5]. In a normal distribution, 68% of the time the price of a stock fall within one standard deviation of the mean. Future prices are within two standard deviations 95% of the time.
The total dispersion of a return is made up by two components. The first is the systemic risk, a type of risk that is persistent and depends on the current status of the economy. Systemic risk derives from economy-wide perils that threaten all businesses [5]. The second component of the total dispersion of an asset is called unsystematic risk, a risk that is specific to any particular asset. It is intrinsic to the company that issues the asset and, therefore, cannot be eliminated.

2.1.4 Portfolio Variance

The portfolio Variance is defined as [5]

$$\sigma_p^2 = \sum_{k}^{N} \sum_{j}^{N} w_k w_j \sigma_k \sigma_j \rho_{k,j} \tag{2.7}$$

where $\rho_{k,j}$ is the correlation coefficient between the return on asset k and asset j

$$\rho_{k,j} = \frac{\mathrm{C}ov_{k,j}}{\sigma_k \sigma_j} \tag{2.8}$$

$$Cov_{k,j} = \sum_{t=1}^{n} (k_t - \mu_k)(j_t - \mu_j) \tag{2.9}$$

As long as $-1 \leq \rho_{k,j} < 1$, a diversification benefit can occur and an investor can reduce the risk of the portfolio by simply investing in different assets which are not perfectly correlated. If $\rho_{k,j} = 1$, then the risk of the portfolio is simply the weighted average of the standard deviation of the asset in the portfolio, and diversification benefits do not materialize. Even if assets are positively correlated, but not perfectly correlated, diversification still allows for the same portfolio expected return with reduced risk [18]. Unsystematic risk can potentially be eliminated by combining together more assets whose prices do not move exactly together, while systematic risk cannot be avoided, regardless of how much an investor diversifies [5].

2.2 Modern portfolio Optimization with Python

Portfolio optimization is a process that can be carried out in several ways. Mathematically, to find the weights of an efficient portfolio associated with an expected return of $R^* \in \mathbb{R}^+$ at time t , the following problem must be solved:

$$\min_{\mathbf{w} \in \mathbb{R}^{\mathbf{n}}} \quad Var[R_t], \tag{2.10}$$

$$\text{subject to:} \quad E[R_t] = R^* \tag{2.11}$$

$$\sum_{i=1}^{n} w_i = 1 \tag{2.12}$$

$$w_i \geq 0 \tag{2.13}$$

The constraint in Eq.2.12 requires that the sum of the weights of the portfolio sum up to 1, i.e. the portfolio is fully invested. Eq.2.13 is a short-selling constraint, meaning that we are assuming that short-selling is not possible and, therefore, that every weight in the portfolio is positive. Alternatively, we could have set a certain risk level $(\sigma^*)^2$ and consequently maximizing the return for that particular risk level:

$$\max_{\mathbf{w}\in\mathbb{R}^{\mathbf{n}}} \quad E[R_t], \tag{2.14}$$

$$\text{subject to:} \quad Var[R_t] = (\sigma^*)^2 \tag{2.15}$$

$$\sum_{i=1}^{n} w_i = 1 \tag{2.16}$$

$$w_i \geq 0 \tag{2.17}$$

By solving one of these two problems, we can delineate a set of Efficient Portfolios: they offer the highest expected return for any level of risk or, on the other hand, the lower level of volatility for any level of return. For the purpose of this Thesis, we will use Python optimization libraries to find the Efficient Frontier. To illustrate the methodology, we proceed with an example of portfolio optimization in Python.

2.2.1 Data Manipulation and Relevant Assumption

Before to start to collect and analyze historical data, it is important to state the relevant assumptions of the case. First, we will use logarithmic returns because they have statistical advantages against the simple returns (see Eq. 2.2); the multi-period logarithmic return is defined as the sum of the one-period logarithmic returns, while the multi-period simple return is the product of the one-period simple returns, which can lead to computational problems for values close to zero [19]. Additionally, logarithms are widely used in finance and are often a more advantageous and powerful way to look at returns [12]. Second, due to limitations in computational power and for reasons of clarity, we prefer to work with a narrow sample size and over a short period of time. For the illustrative example of this Chapter, we combine together four random stocks quoted in the US to form Efficient Portfolios. These stocks are

- General Electric Company(GE)
- The Walt Disney Company (DIS)
- Starbucks Corporation (SBUX)

- The Boeing Company (BA)

Historical data for all the assets cover the period from 2020/05/01 to 2021/05/01 [20]. In Chapter 4, during the Monte Carlo Simulation, the sample size will be broaden and will include more stocks to make the model more reliable and realistic. The risk free rate is assumed to be the Treasury Yield curve rate for 5 years of 05/05/21 (0.77 %) according to the US Department of The Treasury [17].

2.2.2 Data Manipulation

We start by collecting historical Adjusted Closing Price from Yahoo!Finace [20] and then by computing the percentage change in the Adjusting Closing price from one trading day to the following (Eq. 2.2). We set up a data frame that concatenates the 4 stocks in a table and, for visual purposes, only the Normal Returns of first 5 among the 225 trading days are displayed (Fig. 2.1)

Date	General Electric	Disney	Starbucks	Boeing
2020-05-08	0.029460	0.034006	0.024605	0.037233
2020-05-11	-0.015898	-0.012734	-0.025684	-0.033948
2020-05-12	-0.030695	-0.029786	-0.020034	-0.028625
2020-05-13	-0.035000	-0.015685	-0.009818	-0.029708
2020-05-14	-0.015544	0.029052	0.007742	0.008395

Figure 2.1: Daily Normal Returns

It is well know that the relationship between Normal and Log Returns is given by the formula

$$Normal\ Return = e^{Log\ Return} - 1 \tag{2.18}$$

and therefore

$$Log\ Return = \log(Normal\ Return + 1) \tag{2.19}$$

By using the above formula (Eq. 2.19), we can convert Normal Returns into Log Returns and annualize the Log returns by summing up all the daily Log returns of the 252 trading days. However, the definition of the expected return of a portfolio over any time period is the weighted average of all the simple expected returns of the individual securities (Eq. 2.5). It would be erroneous to actually use Annualized Log Returns to compute the one-period return of the portfolio, since there is no compounding element. We are operating on one-period basis, and we should instead use Annualized Normal Return. Revisiting the formula above (Eq. 2.19), we get

$$Annualized\ Log\ Return = \log(Annualized\ Normal\ Return + 1) \tag{2.20}$$

All the following measures have been computed according to the formulas in Chapter 2. The annualized logarithmic returns and the annualized volatility of the returns of the stocks are displayed below:

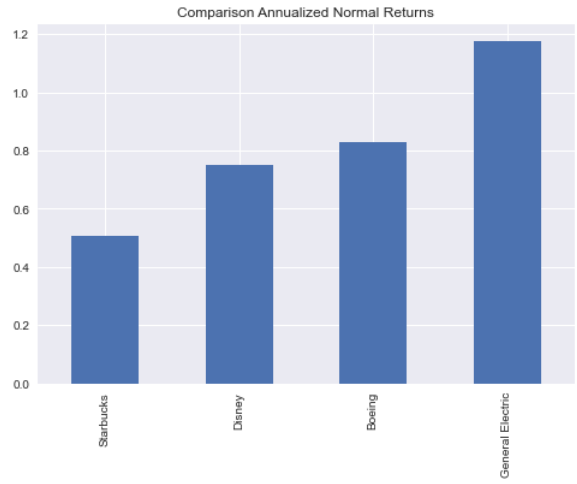

Figure 2.2: A Comparison of the four stocks' Annualized Normal Returns

We can calculate the Annualized Volatility of the 4 stocks by simply using the built-in Standard Deviation function in Python (std()). Then, the result is annualized by multiplying for $\sqrt{252}$. (Figure 2.3)

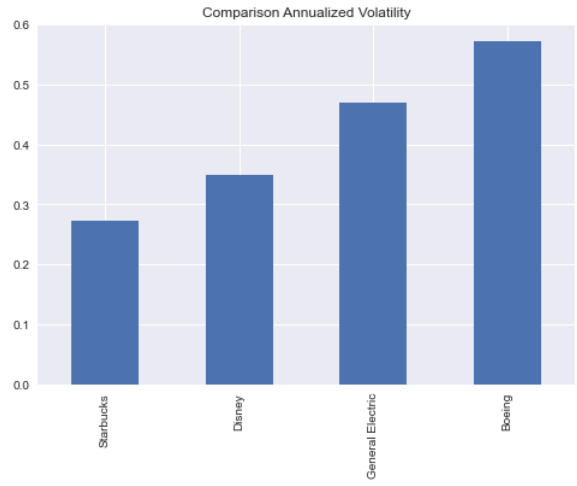

Figure 2.3: A Comparison of the four stocks' Annualized Volatility

We are still missing the Co-variance matrix to see how much the stocks co-move together and to actually compute the portfolio Variance. To fill this gap, we can use the built-in Co-variance Matrix Python function (cov()) and annualize the result by multiplying for 252 (see Fig2.4). Relevant statistical figures of the stocks are displayed below in Fig. 2.5.

	General Electric	Disney	Starbucks	Boeing
General Electric	0.220055	0.076482	0.049651	0.179457
Disney	0.076482	0.121174	0.041532	0.105196
Starbucks	0.049651	0.041532	0.074122	0.087926
Boeing	0.179457	0.105196	0.087926	0.328023

Figure 2.4: Annualized Co-variance Matrix

	Annualized Return	Annualized Volatility	Skewness	Kurtosis	Cornish-Fisher Var 5%	Sharpe Ratio	Max Drawdown
General Electric	1.433412	0.472914	0.430548	5.007893	0.040428	3.030772	-0.296690
Disney	0.862389	0.354641	1.635901	11.271117	0.018996	2.431392	-0.142835
Starbucks	0.561556	0.272045	-0.204266	6.039100	0.026294	2.063770	-0.143490
Boeing	1.159826	0.580417	0.771010	6.710213	0.045828	1.998059	-0.373579

Figure 2.5: A summary of the relevant statistics

2.2.3 Efficient Frontier in the absence of the risk-free asset

Now it is all set for the computation and the plotting of the Markowtiz's Efficient Frontier. Python optimization libraries allow us to quickly solve one of the two optimization problems mentioned in Sec. 2.2. Consequently, different baskets of Efficient Weights can be found. These feasible portfolios allow an investor to maximize the return given any target level of volatility (or, vice versa, to minimize the volatility given any target level of return). Any portfolio that lies above the Efficient Frontier cannot be achieved. Any portfolio that lies beneath the Efficient Portfolio is dominated by at least two portfolios on the Efficient Frontier. The Efficient Frontier represents the only set of portfolios that a Mean-Variance investor should be interested in, because it represents the set of feasible portfolios that have the maximum return for a given level of risk [5].
The pink portfolio at the nose of the curve in Fig 2.7 represents the Global Minimum Variance Portfolio, the most efficient portfolio that provides investors with the lowest level of volatility. When there are only risky assets, Markowitz's Efficient Frontier is the upper side (above the GMV Portfolio) of the envelope of all feasible portfolios.

2.2.4 Efficient Frontier after the introduction of the risk-free asset

When a risk-free asset is introduced, the Efficient Frontier dramatically changes shape and more feasible portfolios are created. In Fig. 2.8, any portfolio that lies on the dotted purple

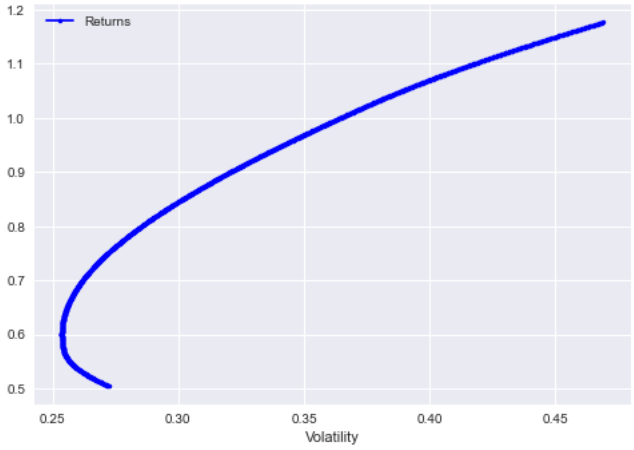

Figure 2.6: Set of Mean-Variance feasible portfolios

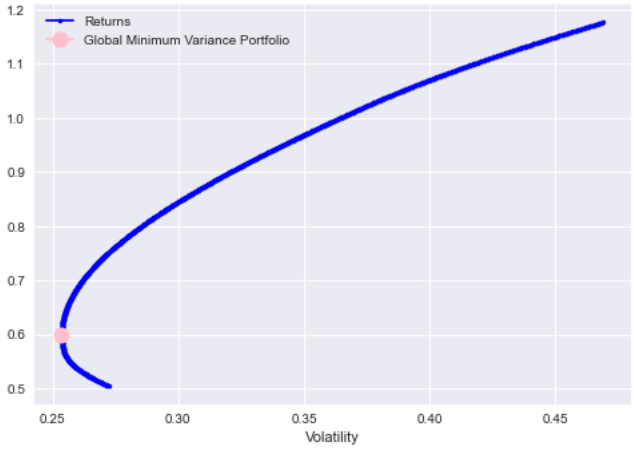

Figure 2.7: The Efficient Frontier without a risk-free asset

line, which goes from the risk-free asset to any risky portfolio, is also a feasible portfolio and a component of the Efficient Frontier [13]. Note that the Efficient Frontier does not continue beyond the Optimal Portfolio because we have imposed a short-selling constraint (Eq. 2.13).

In Fig. 2.8, the purple dotted line that links together the risk free rate and the set of feasible Portfolios is called the Capital Market Line.

The point of tangency (purple dot) is the point where an Efficient Portfolio has the highest Sharpe Ratio, a measures tracked by investors to asses risk-adjusted performance of an investment. The Sharpe Ratio is the most famous measure of risk- adjusted performance and it is defined as the following [5]:

$$Sharpe\ Ratio = \frac{RiskPremium}{Standard\ deviation} = \frac{E[R_p - R_f]}{\sigma_p} \tag{2.21}$$

Figure 2.8: Efficient Frontier

where R_f is the risk free rate. It measures the amount of excess return per unit of risk, measured as volatility; the higher is the Sharpe Ratio, the more efficient is the portfolio. This measure of performance is straightforward to be computed and allows for instantaneous comparison of portfolios' performance. This is why we will use the Sharpe Ratio as the unique measure of portfolio's performance for the purpose of this thesis. However, it also has its limitations. The main drawback is that the Sharpe Ratio does not distinguish between downside and upside potential. Given that one of the main objective of risk management is to avoid only downside risk, the Sharpe Ratio may lead to analyses that are flawed and other ways to gauge the performance of a portfolio should be considered.

The Tangency Portfolio (also known as the Maximum Sharpe Ratio Portfolio) is the portfolio that gives to an investor the highest reward per unit of volatility. It can be easily shown that the Tangency Portfolio contains only systematic risk and no exposure to specific risk. This characteristic is the reason why the Maximum Sharpe Ratio Portfolio is so attractive: by diversifying away specific risk (that is not rewarded by financial markets), the Maximum Sharpe Ratio Portfolio does a very good job at maximizing the reward per unit of risk. According to the Capital Asset Pricing Model, all investors hold the Tangent Portfolio as their Optimal Portfolio, varying only the amount invested in the Optimal Portfolio and in the risk-free-asset according to their personal preferences [5].

The yellow portfolio is a naive Equally Weighted Portfolio, in which each asset has the same weight (25 %). Clearly, this portfolio is not efficient and dominated by at least other

two portfolios that lie on the Efficient Frontier. In this specific case, investors would never hold the Equally Weighted Portfolio (in general, they would never invest in any portfolio that lies beneath the Efficient Frontier) as they could get either:

- Higher return for the same level of volatility
- Lower volatility for the same level of return

All investors optimally choose to hold a combination of the Maximum Sharpe Ratio portfolio and the risk-free asset. In our 4-stocks-model example, the blue dot in Fig. 2.9 is the

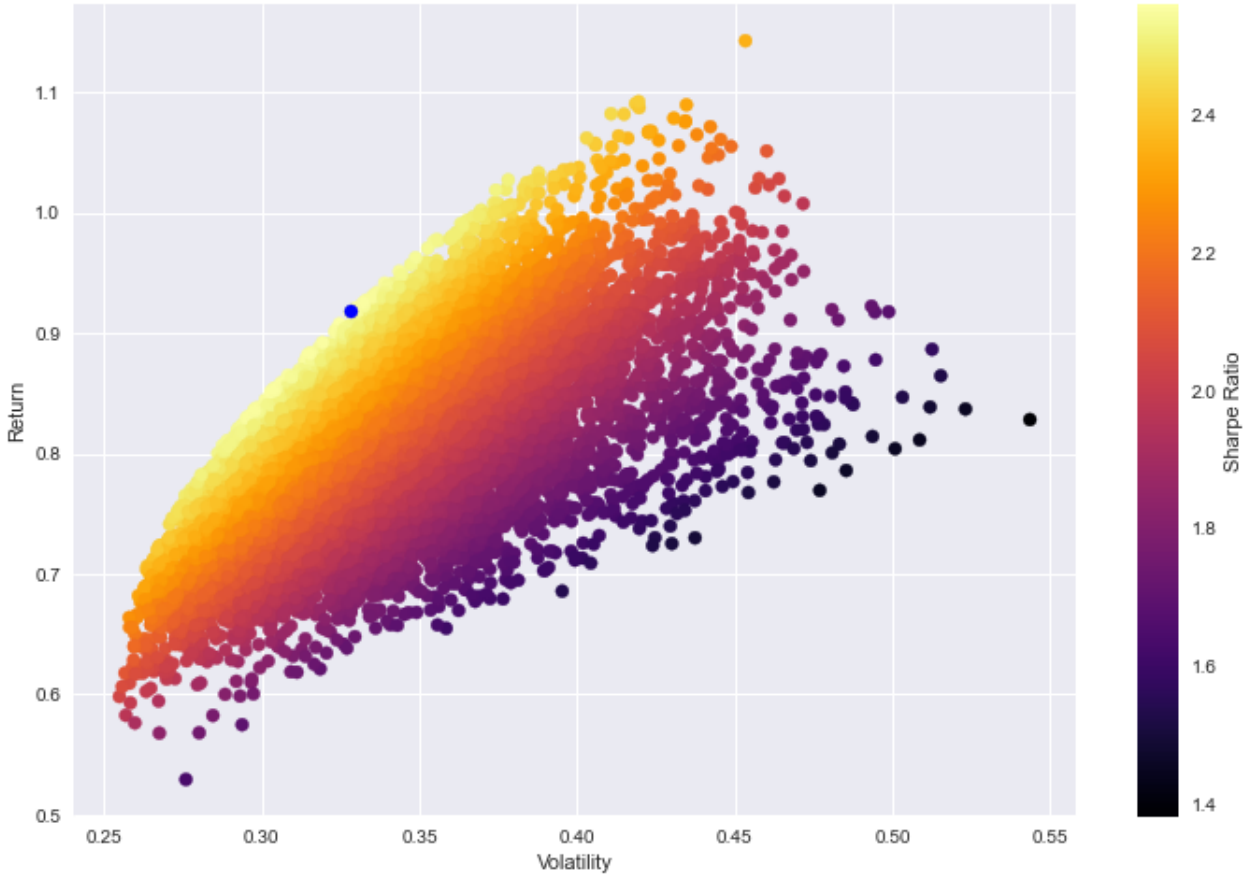

Figure 2.9: The Optimal Portfolio

Optimal Portfolio and has an Expected Return of approximately 87 %, a volatility of 31 % and a Sharpe Ratio is 2.77. The composition of the Optimal Portfolio is the following:

- General Electric 42.89 %
- Disney 29.98 %
- Starbucks 27.07 %
- Boeing 0.6 %

The Sharpe Ratio seems quite impressive. However, we should keep in mind that portfolio performance are biased and altered by the very narrow time-horizon that we imposed at the beginning of this Chapter. The Covid-19 pandemic caused US financial asset evaluations to plummet in March 2020 and consequently to rally in the stock market starting from April 2020. Many profitable and peculiar opportunities were created at that time. It follows that we should not expect these abnormal returns to happen again in the future. Therefore, sample average of returns of a short period of time might not be accurate as a proxy for the expected return and volatility of a portfolio.

2.3 Major drawbacks of MPT

Unfortunately, early applications of Markowitz's optimization technique based on naive estimates of the input parameters have been found of little use because leading to non-sensible portfolio allocations. Despite the fact that MPT is still widely used among the major financial centers all around the world, lately the underlying assumptions of this mathematical framework have been widely criticized. The implementation of the Mean-Variance optimization method is affected by a sampling error because it is based on biased estimations of expected variables obtained from historical data. Moreover, the portfolio weights tend to be extremely sensitive to very small changes in the expected returns estimates. As a result, even a small increase in the expected return of just one asset can dramatically alter the optimal composition of the entire portfolio [11]. In other words, a minor perturbation of the inputs (mean return vector and Co-variance matrix) may result in two completely different portfolios. This is what some experts call the error maximizing nature of Markowitz's analysis, because an estimation error that is fairly minor may cause massive shift in portfolio weights. As stated in Sec. 2.1.3, volatility is the simplest measure of risk under the assumption of normally distributed returns. However, empirical data suggest that price levels do not necessarily follow the pattern of a normal probability distribution [10]. Moreover, this extreme computational simplicity is accompanied by several drawbacks. The most important disadvantage is that variance treats gains and losses symmetrically [14]. As a result, alternative risk measures and better estimation techniques should be taken into account by financial managers to overcome these weaknesses and achieve a better theory-practice fit in portfolio optimization practices.

Chapter 3

The Monte Carlo Simulation and the Efficient Frontier

Chapter 2 starts by analyzing the intuition behind the Monte Carlo simulation and how this powerful algorithm was born. Then, the Random Walk theory is briefly introduced and random different scenarios about stock prices and returns are generated for one of the stocks we analyzed in Chapter 1. This will be propaedeutic to properly run a Constant Proportion Insurance Strategy in Chapter 4. Therefore, the Monte Carlo simulation method and one optimization algorithm are implemented to identify the set of weights that maximizes the Sharpe Ratio of the portfolio composed by the 15 stocks with the highest market cap in the S&P 500 index. Finally, the performances of the Maximum Sharpe Ratio and the Global Minimum Variance Portfolios obtained through the optimization algorithm are compared with those ones derived from Markowitz's analysis.

3.1 History

The Monte Carlo simulation is a method of estimating the value of an unknown quantity using inferential statistics. The archetype of the method was firstly formulated in 1946 by the Polish-American mathematician Stanislan Ulam while he was working on thermonuclear weapons. During that period, Stan Ulam was recovering at home from a serious illness. He had plenty of spare time to be dedicated to a particular type of solitary card game. He was keeping on losing and, really soon, his mathematical mindset made him wonder about

the probability of winning the game. He spent a considerable amount of time in trying to estimate the calculations; however, the combinatorics were just too hard, even for a mathematician of his level. He started to think about an innovative way to solve the problem and seriously took into consideration the possibility of combining together statistical inference and electronic computer techniques. He knew that estimating the probability by using a succession of random operation would have taken several years. Fortunately, Mr. Ulam had really some good friends. He talked to John von Neumann, who is considered as the inventor of the stored program computer, and von Neumann recognized immediately the brightness of his friend's intuition. The underlying idea was that, under certain assumptions, a random sample tends to exhibit the same properties of the population from which it is drawn, according to the Law of Large Number. Together, they were able to implement Ulam's intuition using the first digital computer, the ENIAC, and a new groundbreaking computational algorithm was born under the code-name " Monte Carlo simulation " [7]. In recent years, the Monte Carlo simulation has become an essential tool in the pricing of derivative securities and in risk management [9].

3.2 Random Walk theory

The Random Walk theory states that stock prices are independent of other factors, so the past movements cannot predict future ones. The theory was introduced in 1972 by Burton Malkiel in his book " A Random Walk down Wall Street " and it still represents one of the most controversial theory in finance. The main point of the theory is that stock prices follow a random path and that they are no more predictable than a walking path of someone who is drunk. Statistical research has shown that stock prices tend to follow a random walk with no discernible predictable patterns that investors can exploit. Such findings are now taken as evidence of market efficiency, meaning that market prices are supposed to reflect all currently available information. Only new information will move stock prices, and this information is equally likely to be good news or bad news [3]. Assuming that markets are strong-efficient, it is impossible to beat or predict stock prices given that they already reflect all the available information in the market. As a result, both technical and fundamental analysis are useless.
Asset returns are often assumed to follow a random walk in the financial industry, meaning they are assumed to be independent and identically distributed, with zero serial correlation and a variance proportional to time.

3.2.1 Geometric Brownian Motion Model

The Geometric Brownian Motion Model process has been introduced in 1900 by the French mathematician Louis Bachelier. The Brownian Motion model can be used as a building block for generating scenarios for stock returns. Consider a stock with price P_t at time t and an expected annualized rate of return μ. The purchase price over the next period of time Δt can be decomposed in two parts. The stochastic model for asset returns that we are going to working with is

$$\frac{P_{t+\Delta t} - P_t}{P_t} = (R_f + \sigma\lambda)\Delta t + \sigma\sqrt{\Delta t}\zeta_t \tag{3.1}$$

where:

- $P_{t+\Delta t} - P_t$ is the change in the stock price between t and t+Δt in dollar value
- P_t is the stock price at time t
- R_f is the risk-free rate
- σ is the percentage annualized volatility , i.e. the annualized volatility of the stock index
- λ is the Sharpe Ratio of the stock index
- Δt is an infinitesimal small time period
- ζ_t is a random normal number that represents unexpected events and, therefore, the stochastic part of the Brownian motion model.

In particular, we define μ as the percentage drift, i.e. the annualized expected return of the stock price. In our model, μ is assumed to be the sum of the risk-free rate and product of σ and λ

$$\mu = R_f + \sigma\lambda \tag{3.2}$$

Therefore, Eq. 3.1 becomes

$$\frac{P_{t+\Delta t} - P_t}{P_t} = \mu\Delta t + \sigma\sqrt{\Delta t}\zeta_t \tag{3.3}$$

The first term of the right hand side of the equation represents a predictable part that can be anticipated in advance. It is the expected return of the stock during the infinitesimally small period of time dt. The second term is a stochastic component that is not predictable and represents the random changes in the stock price during the interval of time dt, in

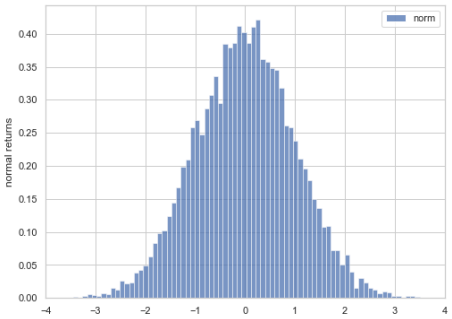

Figure 3.1: Normal distribution of price levels.

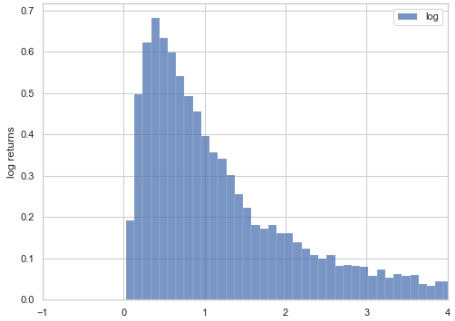

Figure 3.2: Log-normal distribution of price levels

response to external effects such as unexpected news on the stock [6]. Both μ and δ are assumed to be constant. Moreover, GBM assumes the logarithmic change of the stock price at time t to be a normally distributed random variable according to [15]:

$$R_t = \log\left(\frac{P_t}{Pt-1}\right) = \mu + \zeta_t, \qquad \zeta_t \sim Normal(\sigma^2, 0) \tag{3.4}$$

Price levels, instead, are assumed to be log-normally distributed. This assumption often is justified by referring to the historical positivity and right skewness of stock prices. Moreover, given that stock prices cannot be negative, it is illogical to assume a normal distribution of stock prices (see Fig. 3.2).
We can use the Geometric Brownian motion process to construct even more complex asset returns models with time-varying parameters. In reality, interest rates volatility and the Sharpe Ratio change over time. Therefore, the risk-free rate,the volatility index and the Sharpe Ratio used in Eq. 3.1 should not be regarded as constants but as time-varying quantities. In order to have more realistic stochastic scenarios, we would be better-off by allowing those parameters to change over time. However, this augmented time-varying Brownian Motion model falls outside the purpose of this thesis and will not be analyzed.

3.2.2 Random Walk Generation of Asset Prices

By using the model specification described previously in this Chapter, we can run random trials of stock prices. The stock price of Disney on 5/05/2021 was $ 181.5 according to Yahoo!Finance. As illustrated in Fig. 2.5, the drift μ (expected annualized return) of Disney is estimated to be 86.2 %. The annualized volatility was estimated to be 35.5 %. Below is a chart of the outcome where each time step (or interval) is one trading day and the series runs for a whole years (252 trading days). This tool will be deeply exploited in Chapter 4 when we will run the Constant Proportion Portfolio Insurance.

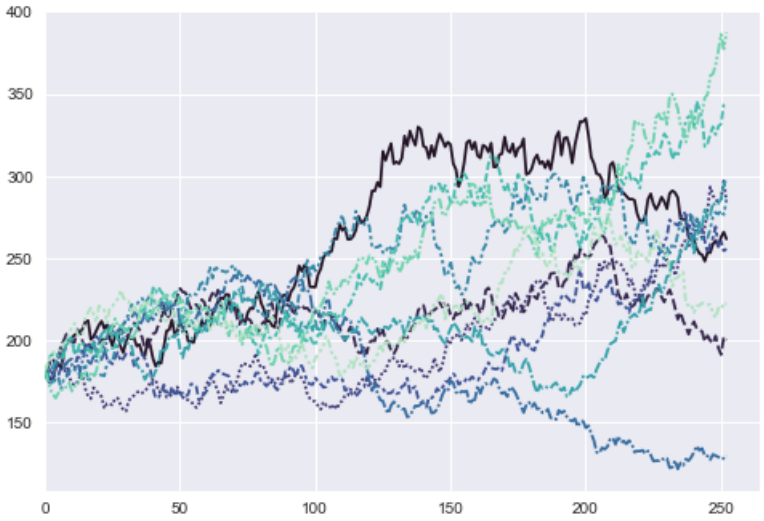

Figure 3.3: Random Generation of the stock prices for Disney over 1 year starting form 5/05/2021

The simulation has produced a distribution of hypothetical future outcomes. Fig 3.3 represents the evolution of Disney's stock price using a Geometric Brownian motion model; each line illustrates one possible way in which the stock price may evolve over time.

3.3 Portfolio Optimization through a Monte Carlo simulation

The term Monte Carlo simulation is usually associated with the process of modeling and simulating a system affected by randomness [4]. The optimization problem presented in Subsec. 2.2 seems easily solvable when there are few financial assets. Indeed, efficient algorithms for solving the quadratic maximization problem formulated by Markowitz are widely available. No Monte Carlo simulation is actually required. However, the solution of the problem proposed by Markowitz may lead to a composition of a portfolio that, even if is correct from a mathematical standpoint, is economically unreasonable. As we will see at the end of this Chapter, the pure resolution of this problem may lead to extreme portfolio allocations. The Optimal Portfolio obtained by running a Monte Carlo simulation seems to better tolerate sample error in the estimation of the return parameters and to be more efficient in a forward-looking prospect.

3.3.1 Optimization Algorithm

For the purpose of this simulation, we will first analyze historical data of the 15 stocks with the highest market cap in the S&P 500 index and, secondly, formulate an efficient allocation of wealth to maximize the return of the investment while minimizing risk. The stocks are Apple Inc. (AAPL), Microsoft Corp (MSFT), Amazon.com Inc (AMZN), Facebook Inc (FB), Alphabet Inc- A shares (GOOGL), Alphabet Inc- C shares (GOOG),Berkshire Hathaway (BRK.B), Johnson & Johnson (JNJ),Procter & Gamble Company (PG),Visa Inc (V),NVIDIA Corporation (NVDA),Home Depot (HD),Mastercard Incorporated (MA),JP Morgan Chase & Co (JPM),UnitedHealth Group (UNH). Historical data are downloaded from Yahoo!Finance and cover the period between 2015-01-05 and 2020-12-31. We will implement a Monte Carlo simulation to assign random weights (each of which must be negative and the sum of which must be 1) to each of the tickers in our portfolio and plot the returns of each portfolio against its standard deviation. One million portfolios will be generated in this process. The relevant statistical figures of the stocks are displayed below.

.

	Annualized Return	Annualized Volatility	Skewness	Kurtosis	Cornish-Fisher Var 5%	Sharpe Ratio	Max Drawdown
AAPL	0.322121	0.296612	-0.324598	9.847772	0.028721	1.085898	-0.401615
AMZN	0.481881	0.307529	0.520538	9.270472	0.024879	1.566846	-0.365043
BRK-B	0.076390	0.208193	-0.350152	16.023968	0.019102	0.366770	-0.319189
FB	0.231469	0.320190	-0.777574	17.199030	0.030790	0.722818	-0.457374
GOOG	0.223389	0.268257	0.223549	12.276823	0.022735	0.832629	-0.327356
GOOGL	0.221084	0.267083	0.144074	12.522767	0.022945	0.827660	-0.328596
HD	0.197398	0.250287	-1.915085	36.495978	0.022050	0.788565	-0.416634
JNJ	0.100421	0.193952	-0.567738	14.264000	0.018831	0.517607	-0.290193
JPM	0.156406	0.292908	-0.093705	18.548718	0.024464	0.533873	-0.472968
MA	0.277163	0.279927	0.036489	14.586557	0.023719	0.990017	-0.435134
MSFT	0.323164	0.278116	-0.219111	14.171716	0.024823	1.161864	-0.311481
NVDA	0.732803	0.453189	0.024894	14.178530	0.038120	1.616924	-0.594599
PG	0.107700	0.194423	0.185411	17.300981	0.015546	0.553791	-0.255880
UNH	0.250995	0.280009	-0.510044	18.851787	0.024945	0.896273	-0.396146
V	0.228910	0.254910	-0.149350	14.639393	0.022490	0.897881	-0.388694

Figure 3.4: A summary of the relevant statistics

Once we have gathered all the relevant information and put into code the equations defined in Sec. 2.1, we can generate random portfolios by initializing one million random arrays that will constitute the weights of the different portfolios. We require that all the weights must be positive and that they must add up to one, according to the constraints illustrated in Eq. 2.12 and Eq. 2.13. The mechanism of this Monte Carlo simulation is to generate

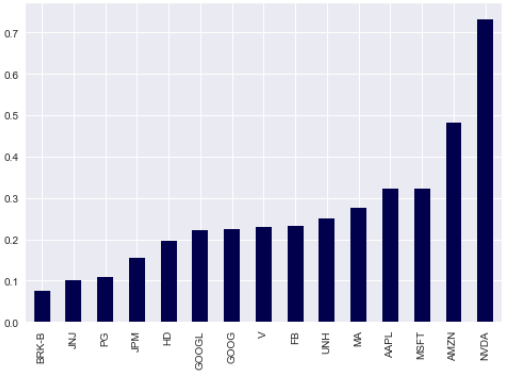

Figure 3.5: A comparison of the annualized logarithmic returns

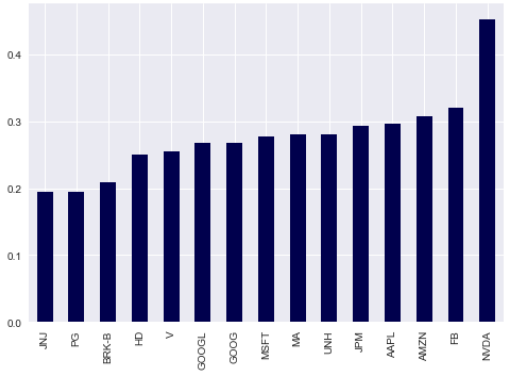

Figure 3.6: A comparison of the annualized volatilities

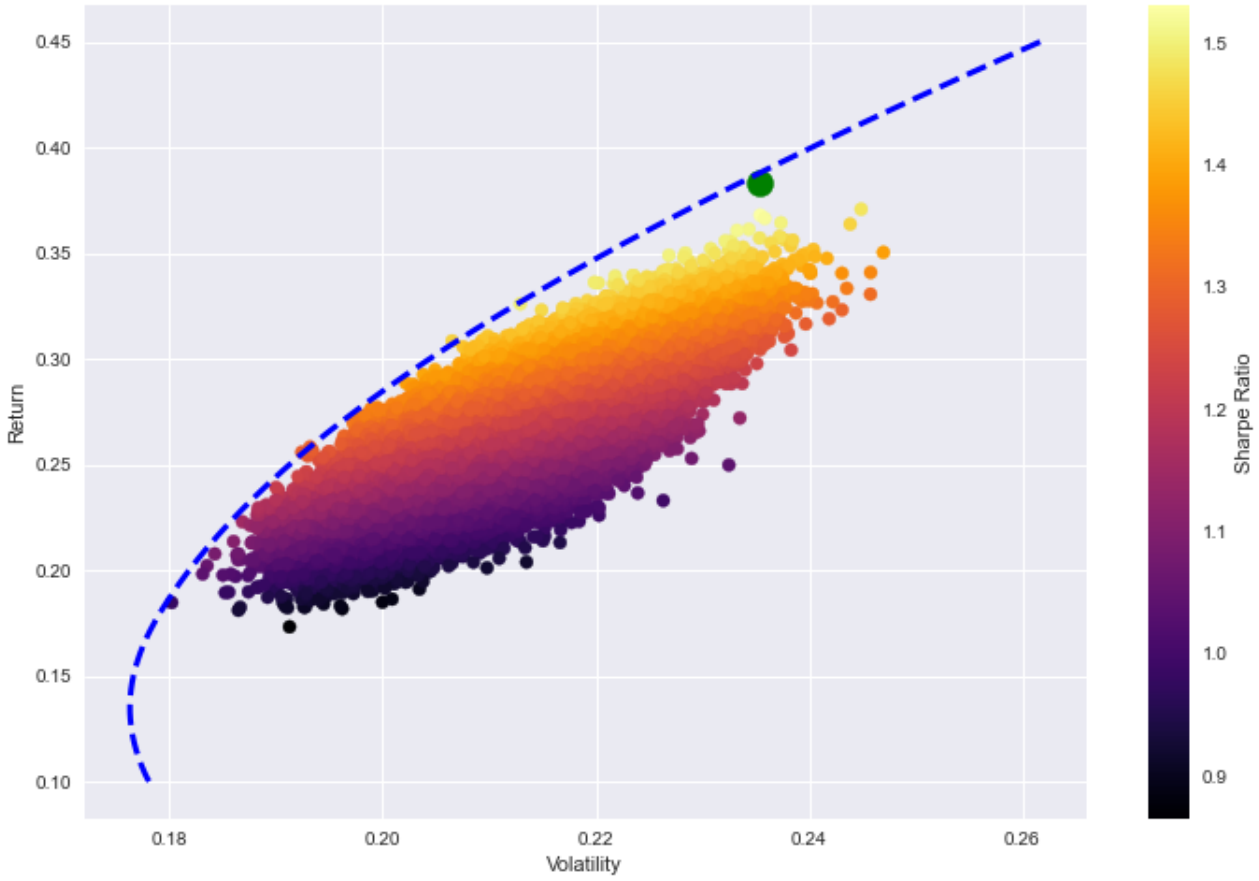

Figure 3.7: The Efficient Frontier and the Maximum Sharpe Ratio Portfolio

one million different scenarios while keeping track of portfolio's return and volatility in every scenario. Then, portfolios with different Sharpe Ratios are plotted in different colors. The blue dotted line in Fig. 3.7 represents the Efficient Frontier, while the green dot is the Maximum Sharpe Ratio Portfolio. This portfolio is invested according to the the following proportions: 14 % in AAPL , 18.9 % in MSFT , 0.7 % in AMZN, 5.3 % in FB, 3.2

% in GOOGL, 1.4 % in GOOG, 3.5 % in BRK.B, 4.2 % in JNG, 3.5 % in PG, 3.7 % in V, 12.6 % in NVDA, 15 % in HD, 2.4 % in MA, 10 % in JPM and 1.5 % in UNH. It provides the investor with a return of 36.8 %, volatility of 23.5 % and a Sharpe Ratio of 1.53. It is the portfolio with the highest reward per unit of risk. The result achieved through the Monte Carlo simulation looks reasonable both in the composition of the weights and in the graphical representation illustrated in Fig. 3.7.

By minimizing or maximizing analytically the problem illustrated in Eq. 2.10 and Eq. 2.14, unreasonable portfolio weights are generated because of the lack of robustness of the Markowitz's model. In our case, the pure mathematical resolution of the maximization problem would suggest to invest 96 % of the initial capital in just 2 stocks, Amazon and NVIDIA, which have the 2 highest expected returns and are among the most volatile stocks. The portfolio obtained theoretically has an expected return of 56.7 % and a volatility of 31.1 %, leading to a Sharpe Ratio of 1.82, which is approximately 0.3 higher than the one obtained by running our Monte Carlo simulation. Such portfolio allocation is biased by parameter estimates of the expected returns of the most high-performing stocks and would make lose an investor the benefits of diversification. Even a estimation error of 1 % in the expected return of a stock may dramatically alter the composition of the portfolio imposed by Markowitz.

Asset Managers tend to avoid the sample-based expected return parameter estimates, which are very noisy and not very reliable, while they tend to engage in estimation of risky parameters that are usually much easier to obtain with a good degree of accuracy. Therefore, professionals implement Markowitz Analysis by focusing more on the Global Minimum Variance Portfolio, instead of the portfolio which maximizes the Sharpe Ratio. This is mainly due to two reasons. Firstly, because of all the limitation of Sharpe Ratio as a measure of portfolio performance illustrated in Sec. 2.2.4. Secondly, the GMV is the portfolio that is the least sensitive to errors in parameter estimates. Since it requires no expected return estimates, it is only sensitive to errors in risk parameter estimates [13]. The estimate for the Global Minimum Variance Portfolio obtained by running our Monte Carlo simulation is more reliable than that one about the portfolio with the highest Sharpe Ratio. By looking at Fig. 3.8, we can see that theoretical GMV Portfolio (red dot) should have a return of 13.5 % and a volatility of 16.4 %; instead, the GMV Portfolio derived from the simulation (grey dot) has the the exact same return with only 1.5 % more volatility if compared to the red dot.

By assigning random weights to each of the tickers in the portfolio and plotting the performance measure of each portfolio, the Portfolio Manager may spot a more diversified Max Sharpe Ratio Portfolio and avoid the extreme sensibility of Markowitz's model, which may lead to biased portfolios with strong positive allocation in some assets and strong negative

allocation in others. The Monte Carlo Simulation seems not to be as sensible to sample error as the Markwowitz formulation, resulting in a well-diversified Portfolio that looks more reasonable in terms of diversification and out-of-sample performances. By summing

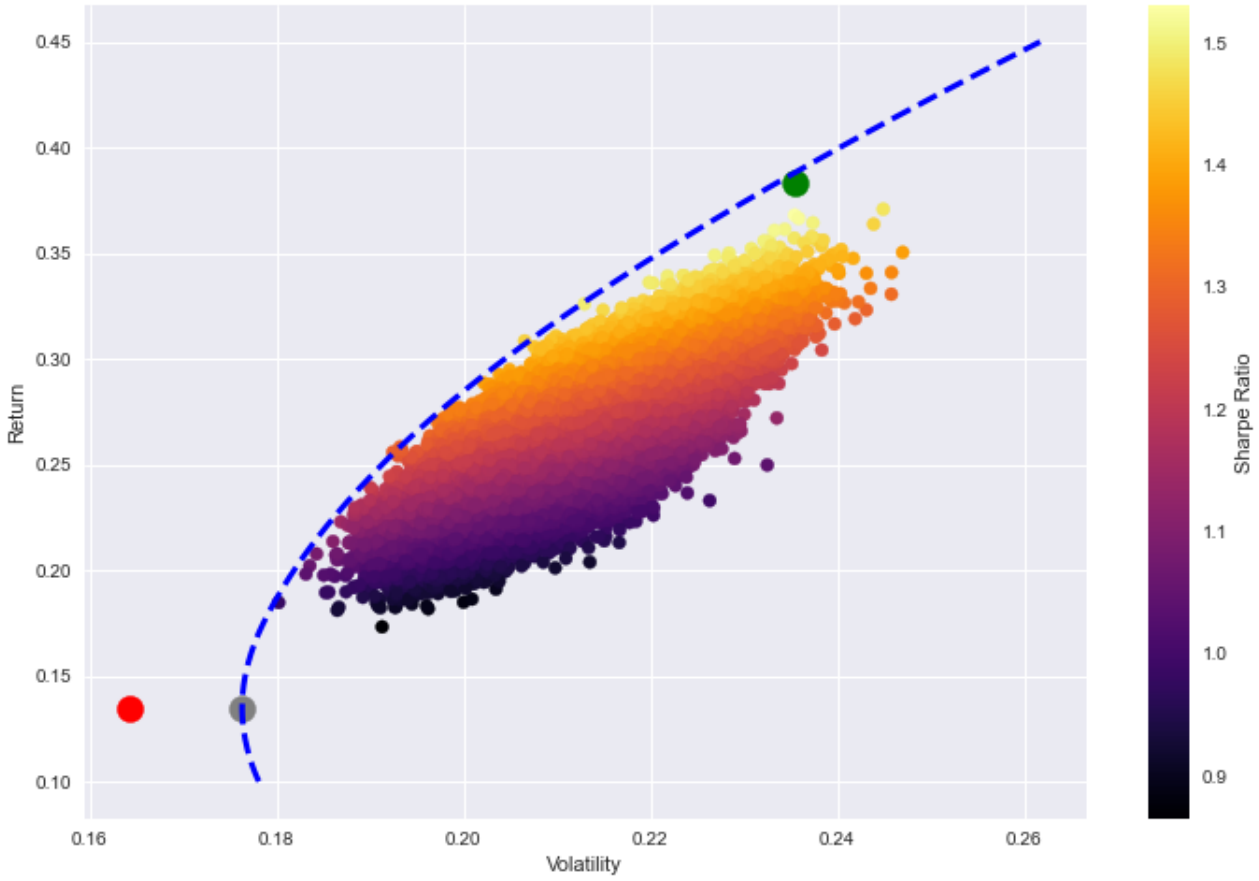

Figure 3.8: The Efficient Frontier and The Global Minimum Variance Portfolios

up, we conclude that Markowitz's Analysis is extremely attractive in principle, because it allows an investor to build efficient portfolios. But in practice, its applicability is severely limited by the presence of errors in parameter estimates [13]. The alternative portfolios that have been generated by running a Monte Carlo simulation appear to be more robust with respect to sample risk and more reliable in terms of a forward-looking prospect.

Chapter 4

The Constant Proportion Portfolio Insurance as a re-balancing strategy

---

The Constant Proportion Portfolio Insurance is an insurance strategy that consists in a dynamic allocation of capital to a risky and a risk-free asset in order to exploit the upside potential of a certain investment strategy by maintaining a downside protection at the same time. The CPPI strategies have been introduced in 1987 by Black and Jones and allow an investor to generate convex payoff functions without the use of any option instrument. Specifically, the payoffs under CPPI can be reconstructed with a position in perpetual American calls on a dividend-paying security [2].
This chapter offers an approach to portfolio insurance that is really intuitive and does not require the use of any complex mathematical formula.

4.1 The Max Drawdown CPPI and its implementation

We introduce a particular version of the CPPI, the Maximum Drawdown Constraints one, whose focus is to maintain the maximum drawdown below a certain pressure level. The Max Drawdown is a very popular measure of downside risk that measures the maximum loss that an investor could have experienced during a certain time period if she had bought a risky asset at its peak price and sold it at its lowest price; it measures the worst possible

return, that is the maximum loss from a previous high to a subsequent low, of a certain risky investment.
In its traditional formulation, the CPPI implies a two-asset framework and that the wealth of an investor is to be allocated between the risky and risk-free asset.
More precisely, we define:

$$Max\ \ Drawdown\ \ Constraint\ :\ V_t > \alpha * M_t \tag{4.1}$$

where:

- V_t is the value of the portfolio at time t
- 1-α is the maximum acceptable drawdown
- M_t is a running max process that keeps track of maximum value of the portfolio between time 0 and time t

This specific approach is particularly well-suited to meet the needs of pension funds' managers who do not want that the value of their portfolio falls below the floor given by the present value of their liabilities [1].
Our dynamic algorithm will try to protect a floor, set by the investor, that represents the minimum dollar level of a portfolio that the investor is willing to tolerate. Note that the floor value is actually a dynamic figure that changes over time as the value of the portfolio changes over time. At every point in time, the floor value will be the maximum previous peak of the portfolio value multiplied by α.
What the strategy says is very intuitive: at every point in time, an investor is going to allocate to the risky asset a multiple of the difference between the current value of the assets in the portfolio and the floor value. This difference is defined as the " cushion ". The percentage of initial wealth allocated to the risky asset is given by multiplying the cushion by a coefficient *m*. The coefficient *m* is usually a function of the maximum drawdown that an investor is willing to tolerate during the investment time horizon; it represents the amount of leverage that an investor is willing to take, as an higher m indicates more aggressive strategies.
Let's suppose an investor wants to invest $ 100 000 in the portfolio derived in Sec. 3.3.1 and imposes $ 90 000 as a floor and 20% as the maximum acceptable drawdown. Also, let's assume that the multiplier *m* is equal to 5. The cushion is $ 100 000 - $90 000 = $ 10 000. According to her risk preferences, the investor will allocate $10 000 * 5 = $ 50 000 to the risky asset and the $100 000 - $ 50 000 = $ 50 000 to the safe asset at this point in time. In other word, we are imposing that the portfolio value must always be greater

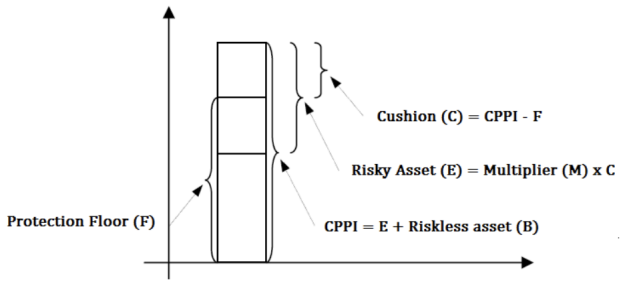

Figure 4.1: Illustration of the Max Drawdown CPPI process

than 80 percent of the maximum value ever reached (not the current value) from the origin of the implementation of the portfolio strategy. The beauty of the CPPI is that if an investor implements it carefully and is willing to re-balance the portfolio extremely often, then nothing can go wrong. As the cushion shrinks to 0 and the the investor gets close to the floor, the portfolio insurance is built in such a way to reduce the allocation to the risky asset and to move capital into the safer asset. However, from a practical point of view, an investor may not willing to trade on a daily basis given the existence of transaction costs. If the investor wants to re-balance the portfolio quarterly, it could happen that, between the two trading dates, the loss in the risky component is so large that she get below the floor before having time to trade. The risk (probability) of breaching the floor because of discrete trading in CPPI strategy is known as gap risk [13].

4.2 The Monte Carlo Simulation and the Max Drawdown CPPI

For the purpose of this thesis, we will run the CPPI for exactly 252 trading days and assume that the risk-free asset is a constant given by the Treasury Yield curve rate for 5 years of 05/05/21 (0.77 %). The risky asset is assumed to be the Tangent Portfolio derived in Sec. 3.3.1, which will follow a Geometric Brownian Motion model. The set of returns for the risky asset over the 252 days will be derived according to Eq. 3.4, using as μ and as σ respectively the expected return and the volatility of the same Tangent Portfolio. We will implement a Monte Carlo simulation that will generate ten different evolution scenarios of the risky asset. The Monte Carlo simulation has proven really useful in testing the outcome of the strategy out of sample and in generating reasonable scenarios for the asset returns by using a Geometric Brownian Motion model. The output of the simulation is an estimate for an unknown quantity (the return of the Tangent Portfolio over the next 252

trading days), which will serve as an input to run the CPPI. Finally, we can generate an asset value history, a risk budget history and a risk weight history. We will get a sense of the whole benefit of the CPPI by comparing the standard wealth index – the evolution of the the investor's portfolio without the application of the CPPI- with the wealth scenario of an investor who has actually implemented CPPI principles. We will call the former portfolio "Portfolio1" and the latter portfolio "Portfolio2". Throughout this Chapter we assume that:

- 1-α is 10 % , i.e. the maximum acceptable drawdown is 10%
- The coefficient m is 3
- The initial wealth of the investor is $1000
- The floor value is 80 % of the current value of the portfolio
- The Cushion is the difference between the value of the portfolio at time t and the floor value at time t. Therefore, the Cushion is initially $1000 - $800 = $ 200
- The Portfolio Manager re-balances the portfolio once every month.

We start by generating the different simulated scenarios of the future wealth for the investor who does not want to apply the CPPI principle discussed so far. Over the next 252 trading day, the portfolio may follow one of these 10 paths: By contrast, we illustrate the

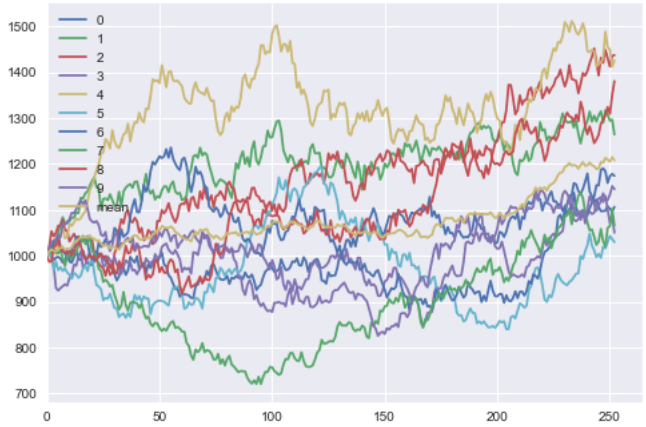

Figure 4.2: Possible paths of Portfolio1

day-by-day progression of Portfolio2 over time in Fig 4.3.
A quick comparison between the relevant summary statistics of Portfolio1 in Fig 4.4 and those of Portfolio2 in Fig 4.5 reveals that CPPI is really effective in limiting downside risk and maintaining upside potential. The average annualized return obtained in scenario1

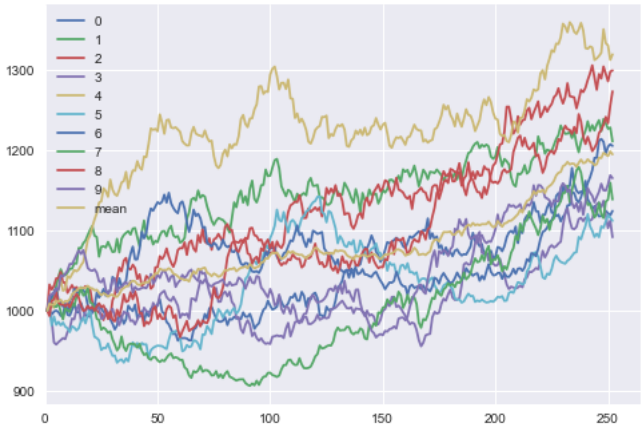

Figure 4.3: Possible paths of Portfolio2

	Annualized Return	Annualized Volatility	Skewness	Kurtosis	Cornish-Fisher Var 5%	Sharpe Ratio	Max Drawdown
0	0.130785	0.208553	-0.130749	2.859786	0.021599	0.626961	-0.108011
1	0.298521	0.230965	0.138178	2.990150	0.022275	1.292363	-0.116461
2	0.481131	0.244941	0.234257	2.744004	0.022809	1.964153	-0.113044
3	0.082653	0.246569	-0.084860	3.449378	0.025414	0.335088	-0.190393
4	0.467864	0.240737	-0.193287	2.889269	0.024226	1.943334	-0.183730
5	0.060823	0.244145	-0.089631	2.871111	0.025441	0.249003	-0.295130
6	0.209097	0.241963	0.139882	2.515173	0.023806	0.864044	-0.287651
7	0.097721	0.241900	-0.004108	2.708031	0.024752	0.403848	-0.309169
8	0.419952	0.237043	-0.039498	2.467604	0.023448	1.771499	-0.115594
9	0.174020	0.222678	0.009527	2.990109	0.022355	0.781353	-0.263359

Figure 4.4: A statistical summary of Portfolio1

	Annualized Return	Annualized Volatility	Skewness	Kurtosis	Cornish-Fisher Var 5%	Sharpe Ratio	Max Drawdown
0	0.120124	0.117184	-0.210472	3.005730	0.012102	1.024829	-0.050818
1	0.214775	0.127488	0.031885	2.864412	0.012361	1.684428	-0.056161
2	0.296681	0.137529	0.167955	2.667553	0.012831	2.157009	-0.057173
3	0.100197	0.131893	-0.182757	3.547260	0.013594	0.759451	-0.080273
4	0.322652	0.126292	-0.108911	2.862774	0.012216	2.554556	-0.077605
5	0.117283	0.117230	-0.137404	3.076809	0.011956	1.000192	-0.115873
6	0.206362	0.117038	0.191413	3.103434	0.010938	1.762937	-0.110995
7	0.135761	0.118337	-0.139410	3.169170	0.011999	1.146986	-0.119699
8	0.296225	0.133061	-0.049825	2.496513	0.012933	2.226008	-0.055938
9	0.161876	0.114073	0.025049	3.179640	0.011124	1.418780	-0.101281

Figure 4.5: A statistical summary of Portfolio2

is approximately 24.7%. However, the Portfolio Manager could have experienced a maximum drawdown of approximately 31 % during the same time period. On the other hand, scenario2 is way more appealing for a risk-averse investors aiming at maximising their upside potential: the CPPI, in this particular case, would have allowed for a maximum drawdown of -11.9 % in the worst case scenario, with an average annualized return of approximately 19.72 % .

Moreover, volatility figures in Portfolio2 are always below 14 % and substantially lower if compared with those in Portfolio1. In particular, let's compare the day-by-day devel-

opment of the two Portfolios. Fig 4.6 illustrates 2 different paths: the black dotted line, representing the average wealth index of Portfolio1, exhibit a more pronounced and fragmented zig-zag pattern with respect to the blue line, which is smoother. Nevertheless, the black dotted line reaches an higher level of wealth at the end of 252 trading days. This is perfectly in line with the CPPI strategy, which sacrifices a reasonable measure of upside potential to protect a floor that, in the absence of transaction costs , would never be broken. The Figure also suggests that the Portfolio Manager experiences more severe drawdowns if the CPPI is not implemented. This is confirmed by Fig 4.7, which compares the maximum drawdown experienced by Portfolio1 vis-a-vis that one experienced by Portfolio2 in the in the worst case scenario (scenario n.7 for both the models). It is worth-nothing that the Max Drawdowns incurred by running the CPPI strategy never breach the 10 % limit by more than 1.6 % (scenario n.5). This is because in our example we are re-balancing every month. However, in a true portfolio management environment, this re-balancing strategy would generate high transaction costs that would seriously harm profitability. That's why most re-balancing strategies envisage a quarterly or annually re-balancing of the portfolio under management. When the volatility in the market is high, as long as Portfolio Managers check and re-balance the portfolio often enough, they might still be able to not incur any violation of the Max Drawdown limit. However, in order to avoid useless transaction costs, an efficient Portfolio Manager usually tries to manipulate the coefficient m. This "aggressivity" parameter plays a crucial in determining the overall benefit of the CPPI in different volatility scenarios. When markets are very volatile, the Portfolio Manager should instinctively use a small coefficient; instead, if the markets are very calm, the Manager might opt for a larger m. In the financial industry no portfolio insurance strategy contemplates a fixed multiplier as our coefficient m. All Portfolio Managers would make sure that m is appropriate for the market conditions that they see going forward [13].

Figure 4.6: A comparison of the Mean Wealth Index of Portfolio1 and Portfolio2

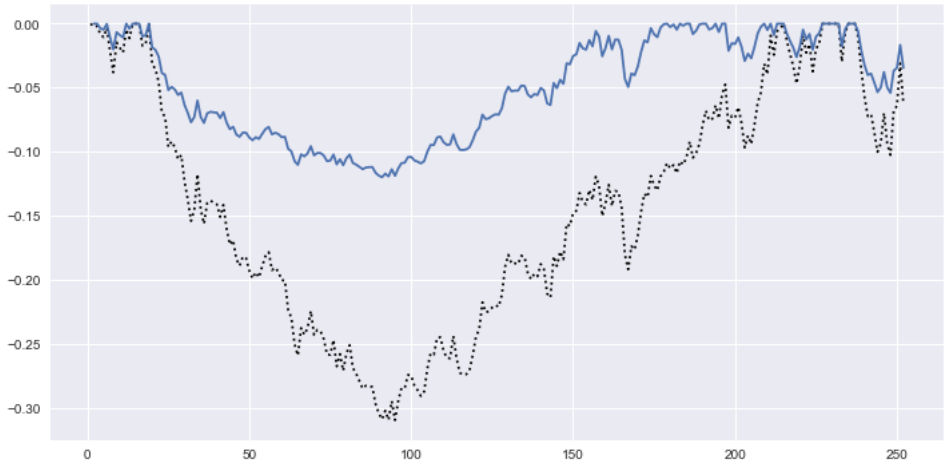

Figure 4.7: Maximum Drawdown comparison

It is interesting to look also at how the composition of the risky and safe weights is altered by the CPPI mechanism all over the trading-year. Initially, the weight allocated to the risky asset is the multiplier m multiplied by the cushion. In our case, the Portfolio Manager allocates 60% of the initial wealth of the investor to the risky asset. During the course of the strategy, the portion of capital allocated to the risky asset in the worst-performing portfolios is reduced to up approximately 30% during the 12 months period. As the value of the portfolio reduces over time, the value of the cushion shrinks as well and consequently the Portfolio Manager decides to allocate an higher portion of capital to the risk-free asset. On the other hand, in the best-performing portfolios, the portion of risky capital fluctuates around its original level, in an interval that goes from 50 % to 60 %. In fact, the mean of capital allocated to the risky asset in each different scenario is 54.8 % .
By generating exactly 1 million different scenarios with our Geometric Brownian Model, it can be seen that the average risky weight among all the different scenarios is approximately 53.6 % .
In conclusion, the CPPI is far more important than people give it credit for because it is the basis for a lot of downside-protection algorithms that are becoming more and more popular into the marketplace [13]. The CPPI is just a core algorithm. No finance-professional implements the CPPI following the step just described in this Chapter. In practice, there are always sophisticated sort of twists that are implemented alongside the standard CPPI. Portfolio Managers would typically impose buffers around their training limits and adjust the CPPI to deal with massive trading costs. Nevertheless, as transaction costs are getting lower, these kinds of algorithms are finding their way into retail accounts. Finally, the outcomes of this strategy really depends on market condition. CPPI is not meant to have lot of downside potential, but rather to have a strong protection floor below which the portfolio value can not fall.

Bibliography

[1] Fischer Black and Robert Jones. Simplifying portfolio insurance. *Journal of portfolio management*, 14(1):48, 1987.

[2] Fischer Black and AndreF Perold. Theory of constant proportion portfolio insurance. *Journal of Economic Dynamics and Control*, 16(3-4):403–426, 1992.

[3] Kane Bodie. Marcus. investments. *Global Edition, 10th, Mc Growth Hill Education*, 1999.

[4] Paolo Brandimarte. *Handbook in Monte Carlo simulation: applications in financial engineering, risk management, and economics*. John Wiley & Sons, 2014.

[5] Richard A Brealey, Stewart C Myers, Franklin Allen, and Pitabas Mohanty. *Principles of corporate finance*. Tata McGraw-Hill Education, 2012.

[6] Abdelmoula Dmouj. Stock price modelling: Theory and practice. *Masters Degree Thesis, Vrije Universiteit*, 2006.

[7] Roger Eckhardt. Stan ulam, john von neumann, and the monte carlo method. *Los Alamos Science*, 15(30):131–136, 1987.

[8] Robert J Elliott and P Ekkehard Kopp. Pricing by arbitrage. *Mathematics of Financial Markets*, pages 1–26, 2005.

[9] Paul Glasserman. *Monte Carlo methods in financial engineering*, volume 53. Springer Science & Business Media, 2013.

[10] Simone Grüninger. Climbing returns and falling meteorites. *Credit Suisse Bulletin,(2)*, 27, 2013.

[11] Martin Haugh. Mean-variance optimization and the capm. *Foundations of Financial Engineering, IEOR E*, 4706:1–7, 2016.

[12] Robert S Hudson and Andros Gregoriou. Calculating and comparing security returns is harder than you think: A comparison between logarithmic and simple returns. *International Review of Financial Analysis*, 38:151–162, 2015.

[13] Edhec Risk Institute. Introduction to portfolio management and construction with python.

[14] Matthias Kull. *Portfolio optimization for constrained shortfall risk: Implementation and IT Architecture considerations*. PhD thesis, M. Sc. Thesis, ETH Zürich, 2014.

[15] Joel Lidén. Stock price predictions using a geometric brownian motion, 2018.

[16] Harry M Markowitz. *Portfolio selection*. Yale university press, 1968.

[17] US Department of The Treasury. Daily treasury yield curve rate.

[18] Iyiola Omisore, Munirat Yusuf, and Nwufo Christopher. The modern portfolio theory as an investment decision tool. *Journal of Accounting and Taxation*, 4(2):19–28, 2011.

[19] Miskolczi Panna. Note on simple and logarithmic return. *APSTRACT: Applied Studies in Agribusiness and Commerce*, 11(1033-2017-2935):127–136, 2017.

[20] Yahoo!Finance. Historical data.

Chapter 5

Appendix

5.1 Python Codes

Some of the following codes have been written with the help of online tutorials and have been revisited by the author for the purpose of the thesis.

```
from math import sqrt
import numpy as np
import pandas as pd
import seaborn as sb
import matplotlib.pyplot as plt
from scipy.optimize import minimize
import scipy.stats as ss
import random
import yfinance as yf
plt.style.use("seaborn")
%load_ext autoreload
%autoreload 2
def check_sum(weights):
    #return 0 if sum of the weights is 1
    return np.sum(weights)-1
import ipywidgets as widgets
```

```
def portfolio_return(weights, returns):

    return weights.T @ returns

def portfolio_vol(weights, cov):
    return (weights.T @ cov @ weights)**0.5
def log_returns(returns):
    log_returns=np.log(1+returns)
    return log_returns
def get_ret_vol_sr(weights):
    weights = np.array(weights)
    ret = portfolio_return(weights,ann_log_rets)
    vol = portfolio_vol(weights,annualized_cov)
    sr = (ret-rf_rate)/vol
    return np.array([ret, vol, sr])

def optimal_weights(n_points, er, cov):
    target_rs = np.linspace(er.min(),er.max(), n_points)
    weights =
    [minimize_vol(target_return, er, cov)
    for target_return in target_rs]
    return weights
def minimize_vol(target_return ,er , cov):
    n = er.shape[0]
    init_guess = np.repeat(1/n,n)
    bounds = ((0.0,1.0),)*n
    return_is_target = {
        'type':'eq',
        'args': (er,),
        'fun': lambda weights, er :
        target_return - portfolio_return(weights, er)
    }

    weights_sum_to_1 = {
        'type':'eq',
        'fun': lambda weights : np.sum(weights) - 1
    }
```

```
def gmv(cov):

    n = cov.shape[0]
    return msr(0, np.repeat(1,n),cov)
def msr(rf_rate ,ann_log_rets , annualized_cov):

    n = ann_log_rets.shape[0]
    init_guess = np.repeat(1/n,n)
    bounds = ((0.0,1.0),)*n
    weights_sum_to_1 = {
        'type':'eq',
        'fun': lambda weights : np.sum(weights) - 1
    }
def sharpe_ratio(log_rets,rf_rate,periods_per_year):
    rf_per_period = (1+rf_rate)**(1/periods_per_year)-1
    ann_log_rets= f_annualize_log_rets(
        log_rets,periods_per_year)
    excess_ret = ann_log_rets - rf_per_period

    ann_vol = f_annualize_vol(log_rets,
                              periods_per_year)
    return excess_ret/ann_vol


def neg_sharpe_ratio(weights, rf_rate, er, cov):
    r = portfolio_return(weights, er)
    vol = portfolio_vol (weights, cov)
    return -(r-rf_rate)/vol

    results = minimize(portfolio_vol, init_guess,
                       args = (cov,), method = "SLSQP",
                       options={'disp':False},
                       constraints=(return_is_target,
                       weights_sum_to_1),
                       bounds=bounds)
    return results.x
```

```
def msr(rf_rate ,ann_log_rets , annualized_cov):

    n = ann_log_rets.shape[0]
    init_guess = np.repeat(1/n,n)
    bounds = ((0.0,1.0),)*n
    weights_sum_to_1 = {
        'type':'eq',
        'fun': lambda weights : np.sum(weights) - 1
    }


def summary_stats(normal_r, rf_rate):

    ann_r = normal_r.aggregate(
        f_annualize_log_rets,periods_per_year=12)
    ann_vol = normal_r.aggregate(
        f_annualize_vol,periods_per_year=12)
    ann_sr = normal_r.aggregate(
        sharpe_ratio,rf_rate=rf_rate,periods_per_year=12)
    dd = normal_r.aggregate(
        lambda r : drwdwn(r).Drawdown.min())
    skew = normal_r.aggregate(skewness)
    kurt = normal_r.aggregate(kurtosis)
    cf_var5 = normal_r.aggregate(gaussian_var,
                                 modified = True)

    return pd.DataFrame({
        'Annualized␣Return': ann_r,
        'Annualized␣Volatility': ann_vol,
        'Skewness':skew,
        'Kurtosis': kurt,
        "Cornish-Fisher␣Var␣5%":cf_var5,
        'Sharpe␣Ratio': ann_sr,
        'Max␣Drawdown': dd
```

```
    })


def drwdwn(return_series: pd.Series):

    wealth_index = 1000*(1+return_series).cumprod()
    previous_peak = wealth_index.cummax()
    drw = (wealth_index-previous_peak)/previous_peak
    return pd.DataFrame({
        'Wealth_Index': wealth_index,
        'Peaks': previous_peak,
        'Drawdown': drw
    })
def skewness(r):
    demeaned_r = r -r.mean()

    sigma_r = r.std(ddof=0)
    exp = (demeaned_r**3).mean()
    return exp/sigma_r**3

def kurtosis(r):
    demeaned_r = r -r.mean()

    sigma_r = r.std(ddof=0)
    exp = (demeaned_r**4).mean()
    return exp/sigma_r**4
def gaussian_var(r , level = 5, modified = False):

    z=ss.norm.ppf(level/100)
    if modified:
        s = skewness(r)
        k = kurtosis(r)
        z= (z+(z**2-1)*s/6+(z**3-3*z)*
        (k-3)/24 - (2*z**3 - 5*z)*(s**2)/36)
    return -(r.mean()+z*r.std(ddof=0))
```

```
def log_returns(returns):
    log_returns=np.log(1+returns)
    return log_returns
def f_annualize_log_rets(log_rets,periods_per_year):
    x=np.mean(log_rets)*periods_per_year
    ann_rets= np.exp(x) -1
    return ann_rets

def f_annualize_log_rets(log_rets,periods_per_year):
    x=np.mean(log_rets)*periods_per_year
    ann_rets= np.exp(x) -1
    return ann_rets

def f_annualize_vol (r,periods_per_year):

    return r.std()*(periods_per_year**0.5)

def plot_ef(n_points, er, cov, show_cml = False,
            style = '.-',rf_rate = 0, show_ew=False,
            show_gmv=False):
    weights = optimal_weights(n_points, er, cov )
    rets = [portfolio_return(w, er) for w in weights]
    vols = [portfolio_vol(w, cov)for w in weights]
    ef = pd.DataFrame({"Returns": rets,"Volatility": vols})
    ax = ef.plot.line(x="Volatility",y="Returns",
                      style=style,color='b')
    if show_gmv:
        w_gmv = gmv(cov)
        r_gmv = portfolio_return(w_gmv, er)
        vol_gmv = portfolio_vol(w_gmv, cov)
        #display GMV
        ax.plot([vol_gmv],[r_gmv], color = 'pink',
                marker ='o', markersize =12,
                label="Global␣Minimum␣Variance␣Portfolio")

    if show_ew:
        n = er.shape[0]
```

```
        w_ew = np.repeat(1/n,n)
        r_ew = portfolio_return(w_ew, er)
        vol_ew = portfolio_vol(w_ew, cov)
        #display EW
        ax.plot([vol_ew],[r_ew], color = 'goldenrod',
                marker ='o', markersize =12,
                label="Equally␣Weighted␣Portfolio")
    if show_cml:
        ax.set_xlim(left = 0)
        w_msr = msr(rf_rate, er, cov)
        r_msr = portfolio_return(w_msr, er)
        vol_msr = portfolio_vol(w_msr, cov)
        #ADD CML
        cml_x = [0, vol_msr]
        cml_y = [rf_rate, r_msr]
        ax.plot(cml_x, cml_y, color='purple',
                marker='o',linestyle = 'dashed',
                markersize = 12,
                linewidth = 2,
                label="Capital␣Market␣Line")
    leg = ax.legend()
    return ax




def gbm(n_years = 10, n_scenarios=1000, mu, sigma,
        steps_per_yea, s_0, prices=True):


    dt = 1/steps_per_year
    n_steps = int(n_years*steps_per_year) + 1

    rets_plus_1 = np.random.normal(loc=(1+mu)**dt,
                                   scale=
                                   (sigma*np.sqrt(dt)),
                                   size=
```

```
                                        (n_steps, n_scenarios))
    rets_plus_1[0] = 1
    x = pd.DataFrame(rets_plus_1)
    x_1 = (x).cumprod()*s_0
    return x_1


def run_cppi(risky_r, safe_r = None, m = 3,
             start = 1000, floor = 0.9, rf_rate=0.0077,
             drawdown = 0.2):

    dates = risky_r.index
    n_steps = len(dates)
    account_value = start
    floor_value = start*floor
    peak = start

    if isinstance(risky_r, pd.Series):
        risky_r = pd.DataFrame(risky_r)
        risky_r.columns = ["R"]

    if safe_r is None:
        safe_r = pd.DataFrame().reindex_like(risky_r)
        safe_r.values[:] = rf_rate/12

    account_history = pd.DataFrame().reindex_like(risky_r)
    risky_w_history = pd.DataFrame().reindex_like(risky_r)
    cushion_history = pd.DataFrame().reindex_like(risky_r)

    for step in range(n_steps):
        if drawdown is not None:
            peak = np.maximum(peak, account_value)
            floor_value = peak * (1-drawdown)
        risky_w = m*cushion
        risky_w = np.minimum(risky_w,1)
        risky_w = np.maximum(risky_w,0)
        safe_w = 1- risky_w
```

```
        safe_alloc = account_value * safe_w
        risky_alloc = account_value*risky_w

        account_value = risky_alloc*(1+risky_r.iloc[step])
        + safe_alloc*(1+safe_r.iloc[step])


        cushion_history.iloc[step] = cushion
        risky_w_history.iloc[step] = risky_w
        account_history.iloc[step] = account_value

    risky_wealth = start*(1+risky_r).cumprod()

    backtest_result = {
        'Wealth': account_history,
        'Risky_wealth': risky_wealth,
        'Risk_Budget': cushion_history,
        'Risky_Allocation':risky_w_history,
        'm': m,
        'start': start,
        'floor': floor,
        'risky_r': risky_r,
        'safe_r': safe_r
    }




    return backtest_result
```

The Monte Carlo simulation applied to find the empirical Maximum Sharpe Ratio Portfolio in Chapter 3 is the following:

```
     np.random.seed(0)
num_ports = 100000
all_weights = np.zeros((num_ports,
                        len(closing_price.columns)))
ret_arr = np.zeros(num_ports)
vol_arr = np.zeros(num_ports)
```

```
sharpe_arr = np.zeros(num_ports)

for x in range(num_ports):
    # Weights
    weights = np.array(np.random.sample(15))
    weights = weights/np.sum(weights)

    # Save weights
    all_weights[x,:] = weights

    # Expected return
    ret_arr[x] = portfolio_return(weights,ann_log_rets)

    # Expected volatility
    vol_arr[x] = portfolio_vol(weights,annualized_cov)

    # Sharpe Ratio
    sharpe_arr[x] = (ret_arr[x]-rf_rate)/vol_arr[x]
print ("Max Sharpe Ratio in the array:{}".format(
    sharpe_arr.max()))
print("Its location in the array:{}".format
      (sharpe_arr.argmax()))

    ### get the allocation of this max:

print(all_weights[sharpe_arr.argmax(),:])
max_sr_ret=ret_arr[sharpe_arr.argmax()]
max_sr_vol=vol_arr[sharpe_arr.argmax()]
min_sr_vol=vol_arr[sharpe_arr.argmin()]
plt.figure(figsize=(12,8))
plt.scatter(vol_arr, ret_arr, c=sharpe_arr,
            cmap='inferno')
plt.colorbar(label='Sharpe Ratio')
plt.xlabel('Volatility')
plt.ylabel('Return')
plt.scatter(max_sr_vol, max_sr_ret,c='blue',
            s=50)# red dot
```

```
plt.show()
```