Undergraduate Thesis

MATHEMATICS DEGREE

PORTFOLIO THEORY: MANAGING BIG DATA

Author: Roberto Rafael Maura Rivero

Tutors: **Dra. Eulalia Nualart**
Economics Department
(University Pompeu Fabra)

Dr. Josep Vives
Mathematics and Computer Science Department
(University of Barcelona)

Barcelona, **19th January 2018**

Contents

0.1 Project

Long and extense is the literature of the portfolio theory. The main goal of it is basically deciding which one is the best possible investment given a set of available assets. During the last decades, there has been great improvement in the way of dealing with the problem and some fundamental changes in the hypothesis taken. The main issue that we will discuss here is that, so far, there is a trade off between the accuracy of the model and the computational time complexity.

In this work, we will discuss some of those famous models that have appeared during history, its particular application and restrictions nowadays, emphasizing in the computational problems derived when working with big data. After that, we will code some of the models and compare them. Finally, we will propose one new model and will compare it with the previous ones.

0.2 Memory structure

The first chapter of this thesis is focused on give a few introductory concepts. I here, we will explain the reasons that justify the need of low risk (standard variation) in the investment world and introduce some computational complexity theory concepts (P, NP, NP complete, NP hard).

The second chapter will focus directly on some of the most popular models throughout history. First we will explain the main concepts and proof the some results in portfolio theory using the Markowitz model (efficient frontier, tangency portfolio, One and Two fund theorems). Then we will present the rest of the models and we will solve the *investors problem* in each of them. We will finish this chapter by presenting a new proposed model that we have called **Clustered Betas**.

In the third chapter, we will use real data and Python code to check some of the results. Also, we will answer questions raised in a research paper ([**BW**]) about closeness between MAD model and Markowitz model. Finally, will check the accuracy of our proposed Clustered Betas model.

In the forth and last chapter, we will present the conclusions of the thesis, next steps and the code used in chapter 3.

0.3 Acknowledgements

This thesis could not have been done without the intervention of many people. First of all, I would like to thanks my two tutors, Professor Eulàlia Nualart and professor Josep Vives, for their grate help during the realization of this work, the orientation, the research

material and the hours valuable advice.

I would also like to thank professor Robert F. Stambaugh from the Wharton School. His classes of finance, the material he shared and the numerous office hours in which he solved my questions and queries have been a great help in order to understand the topics of portfolio theory.

Finally, I would like to thank both my family and friends for being a great support during this last year of college.

Chapter 1

Previous notes

1.1 Goals of the models

First of all, I think is necessary to give an intuitive idea of why the goal of decreasing the variance of the portfolio is something important. There are a few reasons that justify this, and in this short space I am going to explain 2 of them.

1.1.1 Long run investments

First, we have to think how the actual investment works. Once an investor buys a portfolio, the returns are usually reinvested in the same portfolio. Therefore, we have to think that, actually, we are making not only 1 bet, but a series of consecutive bets. We can see that, if that is the case, given 2 investments that are associated with a normal distribution with the same average return but different variance, the expected return at the end of the day can be much higher in the one with lower variance.

The typical example that is given to all undergraduates of finance to internalize this idea is the following:

- Bet A-> 1/2 times you get 120% of your investment and 1/2 of the times you get 100% of your investment
- Bet B-> You get 110% of your original investment for sure.

If you compare the return of obtaining twice 110% (which gives you a return of 121%) versus obtaining 120% once and maintaining 100% the next period, it is very intuitive to see that bet B is better in the long run.

1.1.2 Human behavioral - risk aversion

Secondly, even if we were speaking about investments that take place during only one period and are thought for the short run, numerous studies of human behavior have concluded that people in general, and investors in particular, are risk averse. The definition of risk averse involves some ordering theory and utility theory, and it is not the purpose of this work to explain this other economic field, but intuitively, an investor is risk averse iff given any 2 investments with the same return, she would prefer the one with lower variance. This is studied in more deep in research about Utility Theory.

One of the typical experiments (and easier to do), is offer a random sample of the population the option of earning a certain quantity (e.g. 10 euro) or flip a coin and get the double of that quantity only in the case that the coin shows tails. Numerous research has shown that people in general prefer the secure money rather than the fifty-fifty bet

1.2 Time complexity

During this thesis, we will make constant references to the complexity of the algorithms used to solve the problems. In fact, the whole point of the second chapter of this thesis is comparing models that have been appearing during the last century in terms of accuracy and time complexity. Therefore, I think it is convenient to let a few introductory simple definitions. Notice this definitions are not strictly formal and serve just as a guide to understand posterior topics of this thesis. For more formal definitions, please check [**LE**].

1.2.1 Definitions

Definition 1.1. *A complexity class is a set of problems that can be solved by an abstract machine using an O(f(n)) amount of a resource R, where n is the size of the input.*

Notice that the resource in this context is time, but in other contexts could also mean space.

Definition 1.2. *P (for polynomial time) is the complexity class that contains all decision problems that can be solved using a polynomial amount of time by a deterministic Turing machine.*

Definition 1.3. *NP (for nondeterministic polynomial time) is the complexity class that contains all decision problems for which the cases where the answer is "true" have verifiable proofs in polynomial time by a deterministic Turing machine.*

Definition 1.4. *NP-complete is the complexity class that contains all decision problems T such that T is in NP and any other problem U of the set NP can be reduced to T in polynomial time.*

Definition 1.5. *NP-hard is the complexity class that contains all decision problems V such that exists a NP-complete problem T such that T is reducible to V in polynomial time.*

Informally, one can say that the problem is NP-hard if it is *least as hard as the hardest problems in NP*.

1.2.2 NP-hard algorithms when P = NP

During the following chapter, the reader will learn that some of the more accurate models (ie Markowitz M.V.) need NP-hard algorithms to solve the investor problem (find an optimal portfolio to invest with a required average return minimizing the risk). Notice that this means it does not matter in which of the two scenarios, P=NP or P$\neq$NP, we are. Some of the models would still require too much time.

For this reason, some accuracy needed to be sacrificed for the sake of finding a solution in a reasonable amount of time (P). We will discuss some of the most popular along time. One of the first attempts to simplify it was the CAPM, but we advance that the lack of closeness to reality forced people to continue the search.

Now a days, the most popular models are linear regression models with a few number of factors. This models are easy to compute, and a lot of research is done nowadays focused in finding the key factors with higher R-square and statistical significance.

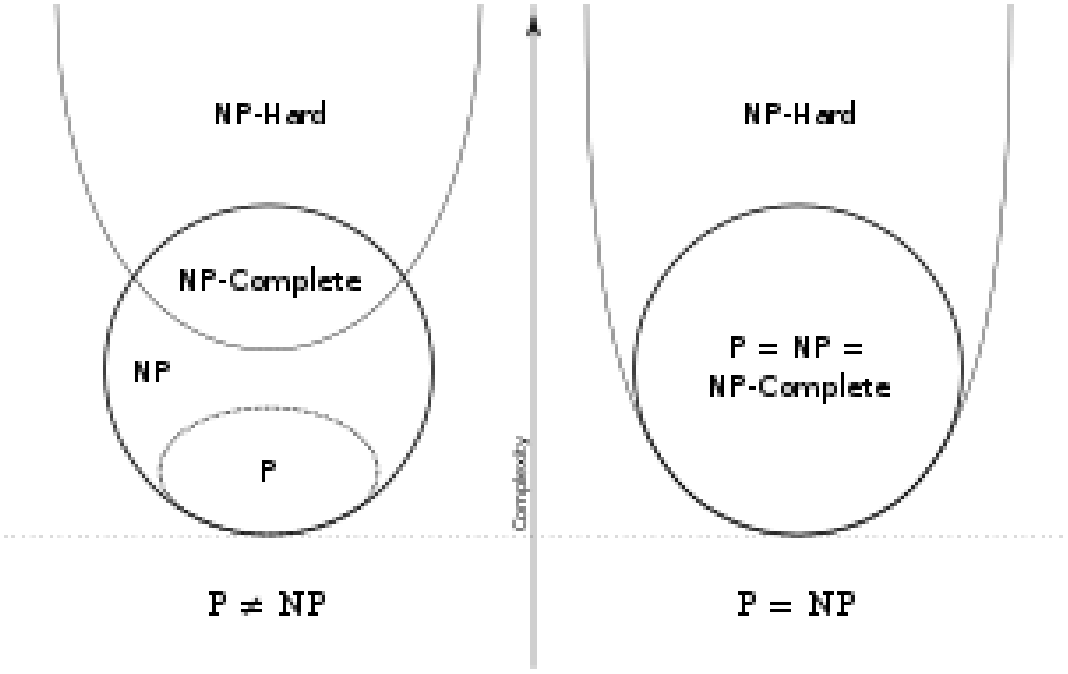

Figure 1.1: This image show the Euler diagram of the two possible scenarios. Notice that even in P=NP, a NP-hard problem will not be solved faster.

Chapter 2

Main models and hypothesis

2.1 Markowitz

The Markowitz model[**MA**] (also called the Mean Variance model or M-V model) assumes that the daily returns of any of the asset of our universe of available assets $\{R_i\}_{i\in I}$ (where $I = \{1, ..., n\}$) is normally distributed with standard deviation $\sigma_i := \sqrt{Var(R_i)}$ and average return $r_i := E(R_i)$, and that there is a certain correlation σ_{ij} between any 2 asset i, j from our universe of shares I (note $\sigma_i = \sigma_{ii}$).

The problem we want to solve is the following: given a minimum desired average return ρ , we want to invest a quantity M_0 among the different asset $i \in I$, investing x_i in each asset, sometimes up to a maximum of u_j, which is the maximum money a investor would place in a single assets, in a way such that we minimize the risk (variance) of the total investment. To simplify the models, we will ignore this last constraint. Further discussion of added constraints is explained later. To sum up, the investors problem can be described in the following way [**LU**]:

Minimize:

$$Min_x(\sum_{i=1}^{n}\sum_{j=1}^{n}\sigma_{ij}x_i x_j), \tag{2.1}$$

subject to

$$\sum_{j=1}^{n} x_j r_j \geq \rho M_0, \tag{2.2}$$

and

$$\sum_{j=1}^{n} x_j = M_0, \tag{2.3}$$

where $x := (x_1, ..., x_n)$.

In real life, traders and investors may sell stocks that they do not own. This operation is called short selling. A summary of the real transaction is the following: the trader borrows a stock, he or she sells it immediately, and buys one again later (after the price drops 10% the following week for example). Then, he or she returns this stock to the original owner, who, of course, will get some interests for the lending. Notice in this model we allow short selling, so a case in which $\exists i \in I$ such that $x_i < 0$ is allowed.

We can restate the problem once more, normalizing it:

Minimize:

$$Min_w(\sum_{i=1}^{n}\sum_{j=1}^{n}\sigma_{ij}w_i w_j), \tag{2.4}$$

subject to

$$\sum_{j=1}^{n} w_j r_j \geq \rho, \tag{2.5}$$

and

$$\sum_{j=1}^{n} w_j = 1, \tag{2.6}$$

where $w_i = \frac{x_i}{M_0}$. Now, we see clearly that the problem does not depend on M_0, the quantity willing to invest.

2.1.1 First attempt solving Markowitz

We will try to solve the problem using Lagrangian multipliers :

$$L = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\sigma_{ij}w_i w_j - \lambda(\sum_{i=1}^{n} w_i r_i - \rho) - \mu(\sum_{i=1}^{n} w_i - 1).$$

After differentiating it with respect to w_i and setting the derivatives equal to 0, we arrive to the following:

$$\begin{gathered}\sum_{j=1}^{n}\sigma_{ij}w_i - \lambda r_i - \mu = 0, \forall i \in I, \\ \sum_{i=1}^{n} w_i r_i = \rho,\end{gathered} \tag{2.7}$$

and

$$\sum_{i=1}^{n} w_i = 1. \tag{2.8}$$

The problem of equations (2.7) is that there is not a closed solution in linear algebra. Therefore, in order to solve it,we have to use numerical analysis.

On a first look, knowing that we will find computational complexity problems with those numerical methods, it might seem that finding a bunch of optimal portfolios for a sequence of ρ is going to be an impossible task.

However, there are some results that will make the work easier. In the following lines we will prove that it is not necessary to calculate the solution portfolio for every single ρ, and that is enough to calculate it for just 2 different ρ_1, ρ_2. The following theorems and proofs are taken from [**LU**].

2.1.2 Two and one fund Theorems

Before stating the theorems of this subsection, it is important to introduce a few new concepts. First of all, we will give a name to those portfolios that solve our problem for a particular given ρ, and then we will introduce the concept of the curve that represent all the solutions of the investor problem for any ρ.

Definition 2.1. (efficient portfolio) *$w = (w_1, ..., w_n)$ is an efficient fund (or efficient portfolio) iff $\exists \rho \in \mathbb{R}$ such that w is the solution of problem (2.4).*

Notation: from now on, we might use the notation $w(\rho)$ to refer to the vector solution of problem (2.4) for a given ρ.

Definition 2.2. (efficient frontier) *If $\{R_i\}$ are all risky investments (ie $\sigma_i > 0 \forall i \in I$) and $R := (R_1, ..., R_n)$, then, we define the efficient frontier (or the efficient portfolio set) as $EF := \{(\rho, Var(w(\rho) \times R)) \in \mathbb{R}^2\}$, where $w(\rho) \times R = \Sigma_{i=1}^{n} w(\rho)_i R_i$*

This set of solutions (the efficient frontier), defines indeed a curve in $\mathbb{R}^2$. To give an idea of how the graph of the efficient frontier looks like, we show in 2.1 a plot of the efficient frontier of random generated portfolios with a bunch of points that represent the individual stocks. In order to understand how it is generated, we will describe the characteristics of the efficient frontier in the case of portfolios generated by 2 assets.

Take two assets i, j from our set of assets I. If the assets are perfectly correlated (ie $\sigma_{ij} = 1$), then, the standard deviation of any portfolio $p = (w, 1 - w)$ (ie investing w in portfolio i and $1 - w$ in portfolio j) generated by combining the 2 assets, has the following variance:

$$\sigma_p^2 = w^2\sigma_i^2 + (1-w)^2\sigma_j^2 + 2w(1-w)\sigma_i\sigma_j = [w\sigma_i + (1-w)\sigma_j]^2$$

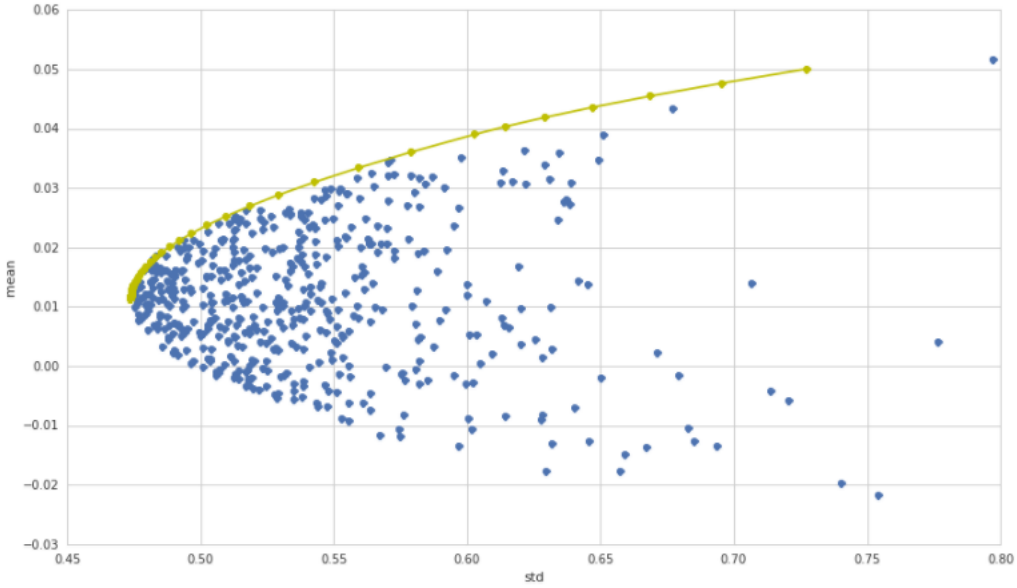

Figure 2.1: Plot of the efficient frontier (in yellow) of randomly generated stocks (blue dots).

Then, the standard deviation is the following:

$$\sigma_p = w_i\sigma_i + (1 - w_i)\sigma_j$$

Therefore, the feasible portfolios are described by a segment, as shown in 2.2 (notice the Y axis indicates the expected return of the portfolio E and the X axis indicates the standard deviation of the portfolio σ).

If, instead, the assets are completely uncorrelated ($\sigma_{ij} = -1$), then:

$$\sigma_p^2 = w^2\sigma_i^2 + (1 - w)^2\sigma_j^2 - 2w(1 - w)\sigma_i\sigma_j = [w\sigma_i - (1 - w)\sigma_j]^2$$

Then, the standard deviation is the following:

$$\sigma_p = |w_i\sigma_i - (1 - w_i)\sigma_j|$$

and the graph of the possible portfolios that are combinations of the portfolios i, j is like the one in the image 2.3.

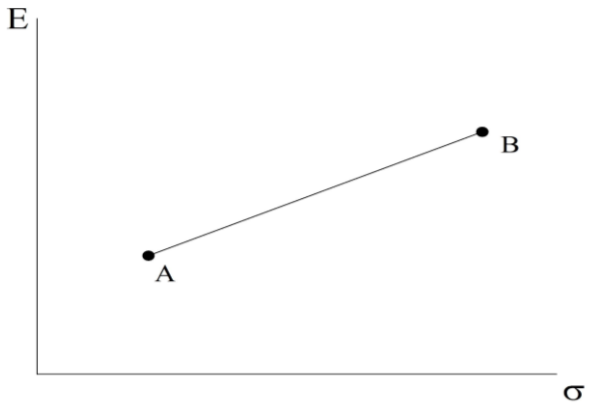

Figure 2.2: Graph of all feasible portfolios generated by 2 perfectly correlated assets.

Notice that, if this is the case, it is possible to obtain a risk free portfolio by investing a quantity $w = \sigma_j/(\sigma_i + \sigma_j)$. Doing this, one would end up with the portfolio that has standard deviation = 0 (the point in the curve of 2.3 that is tangent to the Y axis. In the previous case of perfect correlation, it is also possible to do so, but only by doing short selling one of the assets.

It is now intuitive to imagine the general case, where the standard deviation is the following:

$$\sigma_p = \sqrt{w^2\sigma_i^2 + (1-w)^2\sigma_j^2 - 2w(1-w)\sigma_{ij}}.$$

The curve will lie between the boundaries of the previous extreme cases, given an image like 2.4.

In the general case, with 3 or more linearly independent assets, the set of feasible portfolios stops being just a line or a curve, and can be represented by a convex set (2.5).

Now, we will explain how the theory changes when we introduce a particularly curious asset: the risk-free asset. As is name states, this asset has standard deviation equal to 0. Now we are going to introduce the concept of tangency portfolio. Intuitively it is easy to understand looking at 2.6. To find it, we draw the efficient frontier of the universe of assets I. If we add R_f to that universe, then, given that the risk free asset is uncorrelated to any other asset $i \in I$, the new efficient frontier can be drawn by just a line with the point $(0, R_f)$ of the graph and tangent to the EF in the highest possible point. A more formal definition is presented in the following lines.

Definition 2.3. (tangency portfolio) *If $\{R_i\}_{i\in I}$ are all risk investments (ie $\sigma_i > 0 \forall i \in I$), $EF(\{R_i\}_{i\in I})$ is its efficient frontier of $\{R_i\}_{i\in I}$, and R_f is a risk-free asset, then we will call tangency portfolio to the intersection of both efficient frontiers with and without the Risk Free asset:*

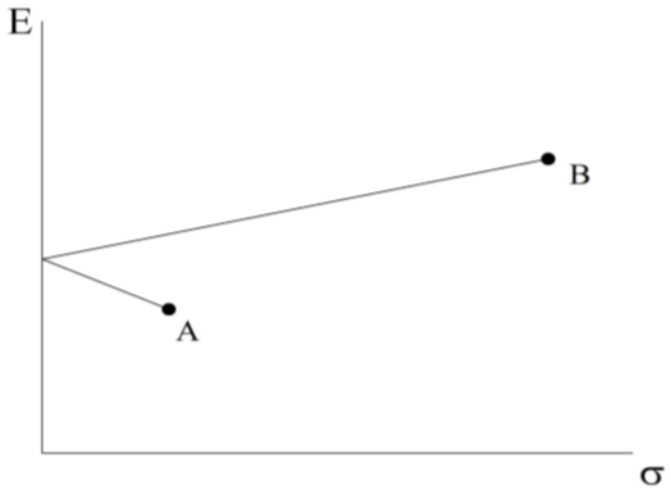

Figure 2.3: The line represent all feasible portfolios generated by 2 perfectly negatively correlated assets.

$$EF(\{R_i\}_{i\in I} \cup \{R_f\}) \cap EF(\{R_i\}_{i\in I}).$$

If we assume that the behavior of the assets is exactly as predicted by the models, then we have the following 2 results:

Theorem 2.4. (Two fund theorem)
$\forall \rho^1, \rho^2 (\rho^1 \neq \rho^2)$ *then* $w := t \times w(\rho^1) + (1-t) \times w(\rho^2)$ *is and efficient portfolio of expected rate of return* $(t\rho^1 + (1-t)\rho^2)$.

Proof. It is enough to check that, if the first 2 portfolios fit into the n+2 linear equations of 2.7, then w will also fit. Let's recall the equations

$$\sum_{j=1}^{n} \sigma_{ij} w_i - \lambda r_i - \mu = 0, \forall i \in I, \tag{2.9}$$

$$\sum_{i=1}^{n} w_i r_i = \rho, \tag{2.10}$$

and

$$\sum_{i=1}^{n} w_i = 1. \tag{2.11}$$

Let's take $\lambda^1, \mu^1, \lambda^2, \mu^2$ from the solutions of ρ^1, ρ^2

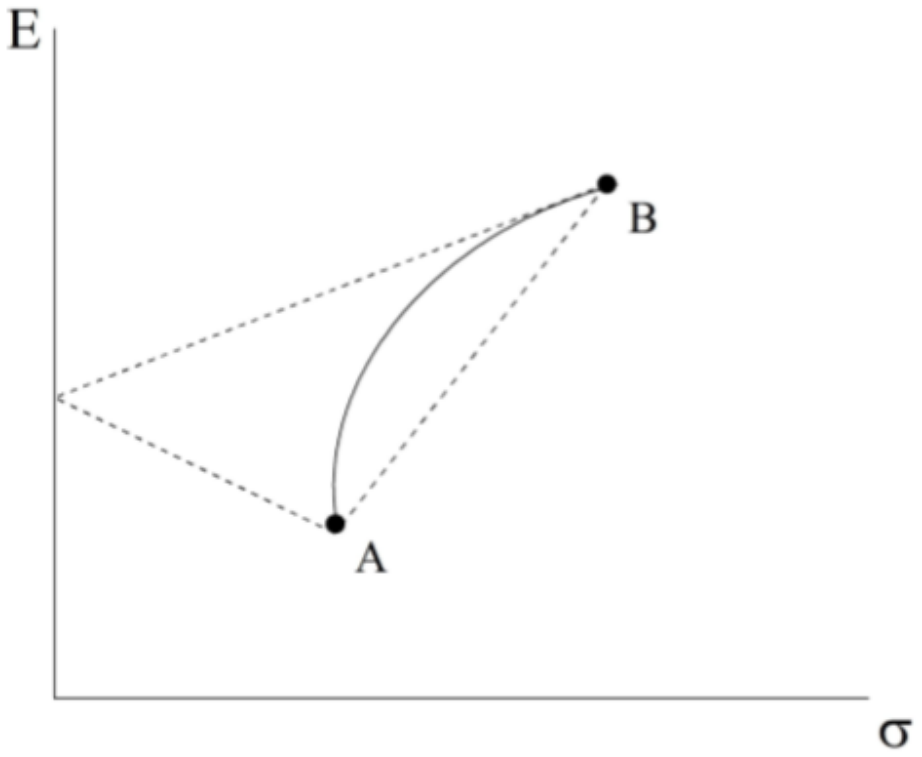

Figure 2.4: This is how the graph looks if the 2 assets are not perfectly negatively nor positively correlated.

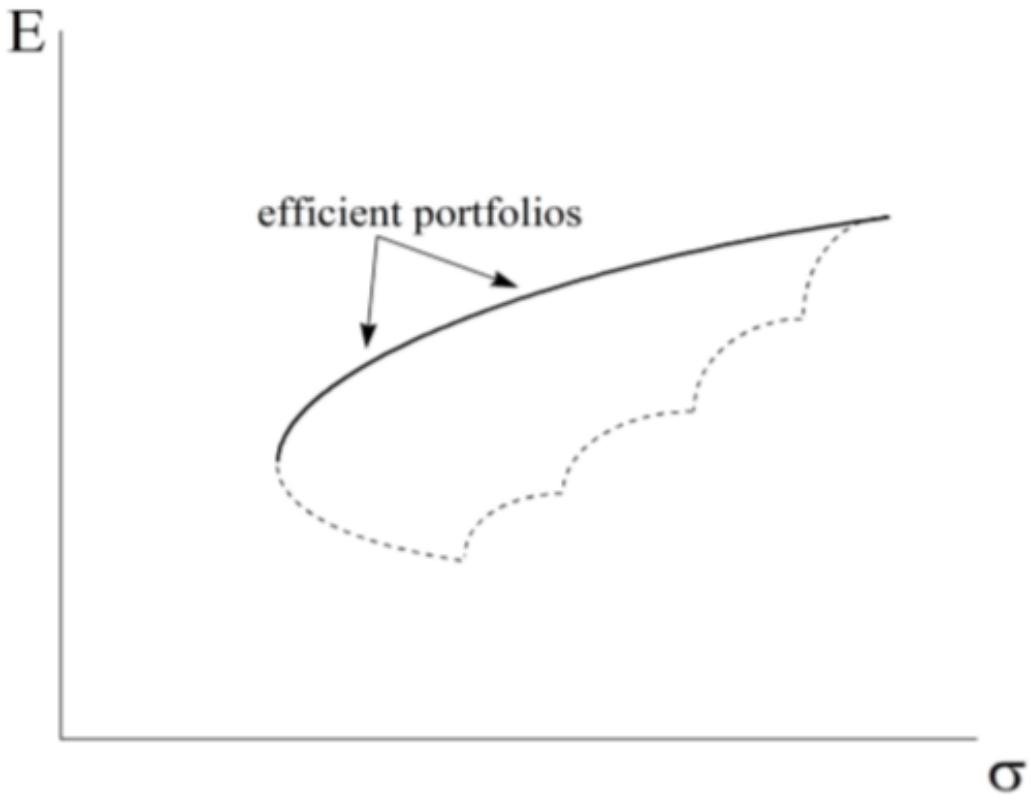

Figure 2.5: The EF is the top part of the frontier of the convex set of all feasible portfolios

$\sum_{j=1}^{n} \sigma_{ij} w_i - \lambda r_i - \mu = 0, \forall i \in I,$

$\sum_{i=1}^{n} w_i r_i = \sum_{i=1}^{n} t w(\rho^1)_i r_i + (1-t) w(\rho^2)_i r_i$

$= t(\sum_{i=1}^{n} w(\rho^1)_i r_i) + (1-t)(\sum_{i=1}^{n} w(\rho^2)_i r_i) = t\rho^1 + (1-t)\rho^2$

$\sum_{i=1}^{n} w_i = \sum_{i=1}^{n} t \times w(\rho^1)_i + (1-t) \times w(\rho^2)_i = 1.$

Finally, notice that, since the 2 solutions make the left side of the equation 2.9 equal to 0, then

$$(w, \lambda^3, \mu^3) := (w, t\lambda^1 + (1-t)\lambda^2, t\mu^1 + (1-t)\mu^2)$$

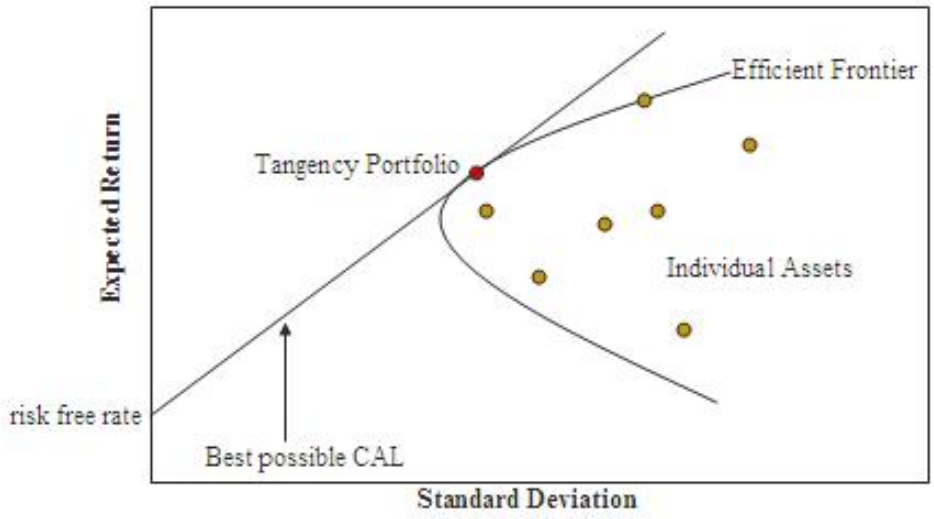

Figure 2.6: New EF with a risk free portfolio and finding the tangent portfolio.

is also a solution of equation 2.9.

□

Notice that, as we allow short selling, t can be any real number. Knowing this result, it is trivial to see that with only 2 different efficient portfolios we can generate the entire efficient frontier. Despite the easy proof of this result, the implications of this in the real world are huge and against the normal intuition. Broadly, this means that in an ideal world, all the investors only have to decide to invest between two funds (portfolios), and the only thing they have to do depending on their required minimum average return is to change the proportion invested in each one.

Theorem 2.5. (One fund theorem) *If there is a risk free asset R_f (i.e., an asset of constant daily return and variance equals to 0) in our universe of assets, then*

$$\exists! F \text{ portfolio of risky assets s.t. } \forall w \in EF(\exists t \in \mathbb{R}(w = tF + (1-t)R_f).$$

Note: in case the reader is wondering if such an asset exists in the real universe of assets, a bond is always given as a good example of what is considered in real life as an asset without risk.

Proof. The proof will first show unity of the previously defined "tangency portfolio". This tangency portfolio is F, indeed. Then, the property that $\forall w \in EF(\exists t \in \mathbb{R}(w = tF + (1-t)R_f)$ will be derived from the 2 fund theorem.

Now, we will describe the process to find the tangent fund. It is obvious that, what we have to do now is maximize

$$tan\theta_w = \frac{r_f - r(w)}{\sigma(w)} = \frac{\sum_{i=1}^{n} w_i(r_i - r_f)}{\sqrt{\sum_{j=1}^{n}\sum_{i=1}^{n}\sigma_{ij}w_i w_j}},$$

where $r(w)$ and $\sigma(w)$ are the return and standard deviation of the portfolio associated at the vector $w = (w_1, ..., w_n)$. We then set all the derivatives of $tan\theta$ with respect to w_i equal to zero for all $i \in I$. Therefore, we have the following expressions with λ as an unknown constants:

$$\sum_{i=1}^{n} \sigma_{ij}\lambda w_i = r_j - r_f, \forall j \in I.$$

Now we substitute $v_i = \lambda w_i$ for each i, and the equation becomes:

$$\sum_{i=1}^{n} \sigma_{ij} v_i = r_j - r_f, \forall j \in I.$$

Now we only have to solve this linear equations for the $v_i's$, and we arrive to

$$w_i = \frac{v_i}{\sum_{k=1}^{n} v_k}$$

([**LU**], 168) Summarizing and restating, the fund $F = (w_1, ..., w_n)$ is the tangent fund and the w_i are given characterized in the following way:

$$w_i = \frac{v_i}{\sum_{k=1}^{n} v_k}, i \in \{1, ..., n\}.$$

Where $(v_1, ..., v_n)$ is the solution to the set of n linear equations

$$\sum_{i=1}^{n} \sigma_{ij} v_i = r_j - r_f, \forall j \in I.$$

With this we have found the tangent portfolio. We now need to know which is the proportion that we need to invest in the risk free portfolio and which one in the tangent portfolio in order to solve our problem (ie obtain a portfolio of return equals to ρ). In order to do so, we only have to calculate the average return of the risky portfolio, which is $r_F = \sum_{i=1}^{n} x_i r_i$. Now, we have to find the proportion t by solving the following equation: $\rho = r_f t + (1 - t) r_F$. □

In order not to complicate the theory with technical details, the literature considers that you can short sell and buy risk free assets with the same return. Otherwise, we would have to change some of the definitions, and every time we speak about optimal portfolios, we would have to do it by using piecewise defined functions (check 2.7).

It is remarkable that the previous 2 theorems have in analogous results in the rest of models presented in this thesis.

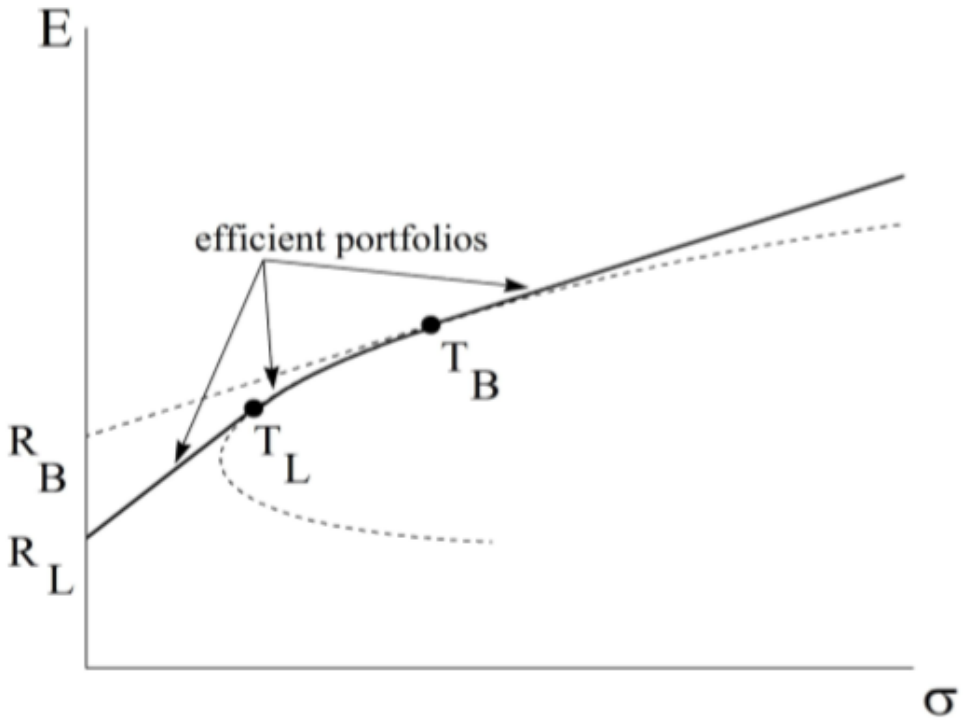

Figure 2.7: This is how the EF would look with a borrowing interest different from the lending interest

2.1.3 Algorithm and time complexity

Despite of the Two fund theorem and the One fund theorem, the stated problem of solving so far is solved with a process of quadratic programming. Quadratic programing is the process of solving a problem of optimization (ie maximization or minimization) of a quadratic function of a certain set of variables subject to a certain set of constrains that are linear . Using matrix notation it can be formulated as follows:

minimize

$$x'Px + q'x,$$

subject to

$$Gx \leq h,$$

and

$$Ax = b$$

where x, q are n-dimensional vectors, b is an m - dimensional vector, h is an l - dimensional vector, A is a $m \times n$ matrix, P is a $n \times n$ matrix, G is a $l \times n$ matrix, for certain $l, m, n \in \mathbb{N}$.

We know that this particular problem has a polynomial worst case complexity. In posteriors and more refined variations of this model, some restrictions are added in order to take into account for the price of the transactions and other details that will be discussed later in this thesis. This increases even more the complexity of the program (one example is the Limited Asset Markowitz model - LAM -, which falls into the class of the NP-hard problems [**CST**]) . This illustrates the main problem when dealing with Big Data: if the number of assets is large enough, then the model of Markowitz (with the necessary added constraints for the particular case of study) is useless in real life because of its time complexity. For that reason, new models had to be created to deal with this problem.

When investors want to use the Markowitz , what is done so far is solving (2.4) for two different ρ with certain software specifically designed for convex optimization. After that, finding any new solution is trivial thanks to the 1 and 2 fund theorem.

2.2 C.A.P.M.

The Capital Asset Pricing Model was introduced in the 60's by different authors independently [**F**], as it is an easy and intuitive simplification of the Markowitz model. The CAPM is a model that, as the Markowitz, assumes that each asset behaves as a normal distribution. However, in this model, the assets are not correlated among themselves as they were in the Markowitz Model. Instead, all assets are a linear regression of one single asset. This gets rid of the computational problem that represented dealing with an entire matrix of correlations.

The new model assumes that each asset is a linear regression of "the overall market". This overall market is represented in the model by a particular asset that is called "the market portfolio". This is a portfolio formed by a weighted sum of all assets of our universe, with the same weights that they represent in the market, assuming that this assets are infinitely divisible. This theoretical concept is formed by ANY asset with value in our universe, which includes all stocks from all markets, all kinds of real state, precious metals or even stamps. However, when this model is implemented, drivers are needed to represent the overall market for obvious reasons, and at the end of the day, people use different index like the S&P500 as the estimators of the market portfolio.

The notation is the following:

- We have a universe of assets $\{R_i\}_{i\in I}$ where $I = \{1, ..., n\}$ plus 2 particular assets called the "risk free asset" R_f (this represents an asset like bonds, that have a fixed return and are considered free of risk in general) and the "market portfolio" R_m
- R_m behaves as a normal distribution s.t. $\sigma_m := var(R_m)$ and $r_m := E(R_m)$
- R_f behaves as a normal distribution s.t. $\sigma_f := var(R_f) = 0$ and $r_f := E(R_f)$ (abusing a bit of the notation, people usually say $E(R_f) = R_f$)
- For every asset $R_i \exists \alpha_i, \beta_i \in \mathbb{R}$ s.t. the following equality holds:
  $R_i - R_f = \alpha_i + \beta_i(R_m - R_f) + \epsilon_i$
  where ϵ_i is a normal distribution with $E(\epsilon_i) = 0$ and a certain variance σ_i(this ϵ_i adds to the model what we call "the idiosyncratic risk").
- $\forall i, j (i \neq j)$ then $covariance(\epsilon_i, \epsilon_j) = 0$

  Fixing our attention in the risk free asset and the market portfolio, notice that $\alpha_m = 0, \beta_m = 1$ and that $\alpha_f = R_f (= E(R_f)), \beta_f = 0$.

  Now, it is easy to deduce the following formulas:

- Expected return of any asset i : $E(R_i) - R_f = \alpha_i + \beta_i(E(R_m) - R_f)$
- Variance of any asset i : $\sigma_i^2 = \beta_i^2\sigma_m^2 + \sigma^2(\epsilon_i)$

2.2.1 Further assumptions and consequences:

All investors **[A]**:

- Aim to maximize economic utilities (asset quantities are given and fixed). Are rational and risk-averse.
- Are broadly diversified across a range of investments.
- Are price takers, i.e., they cannot influence prices.
- Can lend and borrow unlimited amounts under the risk free rate of interest.

- Trade without transaction or taxation costs.
- Deal with securities that are all highly divisible into small parcels (All assets are perfectly divisible and liquid).
- Have homogeneous expectations.
- Assume all information is available at the same time to all investors.

This model with this assumptions helps us to understand in later analysis why α is really close to 0 in most of the stocks. To formally understand why, some Utility Theory is involved. The idea of the explanation is that, if a stock has a positive α, then the investors would buy it, because it increases its utility (he considers that combining the stock with a certain portfolio, the increase in risk is compensated by the increase in expected return). This is because she can maintain the positive alpha while decreasing the risk by shorting a certain quantity of market portfolio, and diversifying with the risk-free portfolio to get rid of some idiosyncratic risk. To sum up, at the end of the day, there is a higher demand of portfolios/stocks with $\alpha > 0$, and this increases the price, decreasing later returns and decreasing the α to 0.

One of the queries the reader might have right now is that, despite that this models assumes some properties for the individual assets (the existence of α and β), maybe this properties are not inherited by the portfolios. Thus, we present the following lemma to prove that any portfolio p fit perfectly into the model, having indeed a characteristic α_p and β_p.

Lemma 2.6. *For any portfolio p that is a combination of $\{R_1...R_n, R_m, R_f\}$, $\exists \alpha_p, \beta_p \in \mathbb{R}$ st $(R_p - R_f = \alpha_p + \beta_p R_m + \epsilon_p)$ where ϵ_i is a normal distribution with $E(\epsilon_i) = 0$*

Proof. It is enough to prove it for the simple portfolio formed by 2 different assets (i, j). Check that if $R_p = wR_i + (1-w)R_j$ then,

- $\alpha_p = w\alpha_i + (1-w)\alpha_j$
- $\beta_p = w\beta_i + (1-w)\beta_j$
- $\epsilon_p = w\epsilon_i + (1-w)\epsilon_j$

Notice that ϵ_p is a normal distribution because is the sum of normal distributions. Also $E(\epsilon_p) = E(w\epsilon_i + (1-w)\epsilon_j) = wE(\epsilon_i) + (1-w)E(\epsilon_j) = 0 + 0$

□

Now we can use in any portfolio the same notation that we used only for the individual assets.

From here, we can deduce that, in the general case, if we invest $x = (x_1, ..., x_n, x_m, x_f)$, then:

- $E(R_p) - R_f = x'\alpha + x'\beta(E(R_m) - R_f)$
- $\sigma^2(R_p) = (x'\beta)^2\sigma_m^2 + x'\sigma^2(\epsilon)$

where $\alpha = (\alpha_1, ..., \alpha_n, 0, R_f)$, $\beta = (\beta_1, ..., \beta_n, 1, 0)$ and $\sigma^2(\epsilon) = (\sigma^2(\epsilon_1), ..., \sigma^2(\epsilon_n), \sigma_m^2, 0)$

2.2.2 Investors problem and One and Two fund theorems

Again, we want to solve the following problem: given a desired minimum return ρ, we want a vector x that represents the portfolio that has an expected return of at least ρ and a the minimum possible variance.

Minimize:

$$Min_x(\sigma^2(R_p) = (x'\beta)^2\sigma_m^2 + x'\sigma^2(\epsilon)) \tag{2.12}$$

subject to

$$\begin{aligned} \sum_{j=1}^{n} x_j r_j \geq \rho \\ \sum_{j=1}^{n} x_j = 1 \end{aligned} \tag{2.13}$$

where $x := (x_1, ..., x_n)$. Notice that, from here, we can deduce again the one fund and two fund theorem with an analogous proof and similar (if not equal) definitions of the previous concepts. Because of the length constraint, we will omit the proof here and in the rest of the models.

The main difference now is that the approach of this problem can be solved with a linear programing process by rewriting $\sigma^2(R_p) = (x'\beta)^2\sigma_m^2 + x'\sigma^2(\epsilon) = x'(\beta^2\sigma_m^2 + \sigma^2(\epsilon))$. Linear programs are problems that can be expressed in the following way:

minimize

$$c'x,$$

subject to

$$Ax = b$$

and

$$Gx \leq h.$$

Thanks to Leonid Khachiyan **[Kh]**, since 1979 it is known that this kind of problems can be solved in polynomial time. Therefore, this model solves the Markowitz issue with the complexity.

2.2.3 Critics

The main problem of this model is its lack of accuracy. In latter analysis, we will see that, despite that this model has no computational complexity problems, it has a low R-square when analyzing stocks and portfolios. The R-square, represented by $\frac{\beta_i^2 Var(R_m)}{Var(R_i)}$, is the part of the variance that is explained by the prediction of the model.

Moreover, further discussion has been made about if the necessary assumptions of the model are actually true. Just to give a quick overview of some of the problems, we refer to the 2004 Fama and French review of the model [**FF**]:

- The stocks do not behave exactly as normal distributions (see problems with fat tails)
- The model assumes all the potential shareholders have access to the same information and agree in all the information of any asset
- The model assumes that the information obtained by the shareholdes is true
- It assumes risk aversion of the investors. Discussion has been made about the possible existence of stock traders - casino gamblers like (ie, risk seeking investors)
- There are transaction costs and taxes, which are not taken into account by the model
- If the investor is big enough, his order of buying/selling could cause a variation in the price of the stock. Again, this is not considered in the model.
- Stocks are not infinite indivisible. You may not be able to buy half a share.
- CAPM may not be empirically testable because of the definition of market portfolio.
- Empirical tests show market anomalies like the size and value effect that cannot be explained by the CAPM. This are later included in variations of the model like FF3factor model.

Notice that some of the assumptions are also made in the Markowitz model. As the reader might deduce from previous comments, this problems are tried to be solved in more detailed posterior models. Despite that it is not the goal of this thesis to discuss them because of the extensive number of more detailed models, we encourage to check more precise versions of the models here presented that take into account different added restrictions (see as one example [**CST**]).

Despite the stated problems, CAPM is still taught and used now a days for various reasons. First of all, it is easy to understand and manipulate. Therefore, in finance courses of many business schools, this is use to give a first approach and the main intuitive ideas of the market. Secondly, when banks and other institutions explain to their investors the main characteristics of their financial products, they usually describe them focusing on the "high alphas and low betas". This is because many of the future buyers of the products do not seek to enter in complicated calculations. Many people even rely on more naive analysis of the assets based on a simple rate (see Moody's, Standand & Poors or Fitch rating tiers).

2.3 M.A.D.

M.A.D. is another alternative to Markowitz M-V model. It was first proposed by Konno and Yamazaki in 1992 [**KY**], and in this model, the normality of the stock returns is not assumed. Using this approach, the concept of risk is reflected by another driver different from the variance or the standard deviation: the mean absolute deviation ($MAD = E[|R_p - r_p|]$, where r_p is the expected return of the portfolio $E(R_p)$).

First of all, we will start with one of the first results of the paper in which this model was presented:

Theorem 2.7. *If $\mathbb{A} = \{R_1, ..., R_n\}$ are multivariate normally distributed, then, for all portfolio $R_p = \Sigma_{i=1}^n x_i R_i$ combination of the universe of assets $\mathbb{A}$:*

$$E[|R_p - r_p|] = \sqrt{\frac{2}{\pi}}\sigma(R_p),$$

where $\sigma(R_p)$ is the standard deviation of the portfolio.

Proof. Let $(r_1, ..., r_n)$ be the means of $(R_1, ..., R_n)$ and also let $(\sigma_{ij}) \in \mathbb{R}^{n\times n}$ be the covariance matrix of $(R_1, ..., R_n)$. Then $\sum_{i=1}^n x_i R_i$ is normally distributed [**R**] with mean $\sum_{i=1}^n x_i r_i$ and standard deviation

$$\sigma(R_p) = \sqrt{\sum_{i=1}^n \sum_{j=1}^n \sigma_{ij} x_i x_j}.$$

Therefore,

$$E[|R_p - r_p|] = \frac{1}{\sqrt{2\pi}\sigma(R_p)} \int_{-\infty}^{\infty} |u| exp\{-\frac{u^2}{2\sigma^2(R_p)} du\} = \sqrt{\frac{2}{\pi}}\sigma(R_p).$$

□

The implications of this simple result are strong. This little theorem has just proven that if we are in the case in which the behavior of the portfolios is actually a normal distribution (which, again, is not assumed in this model), using either MAD or the standard deviation as a measure of risk (ie, solving the investors problem using the Markowitz model or the MAD model) is equivalent. However, we have to be careful here. This does not mean that using either model to solve the investor problem will give us the same solution. This is because, as the reader might have already deduce, the stocks do not behave *exactly* as normal distributions. Therefore, despite that we expect the results to be close, they might be different, as we will check later.

Now, the notation that we will use in this chapter is the following:

- $\{R_1, ..., R_n\}$ is the universe of assets, that **do not necessary behave as normal distributions**.

- $\mathbb{T} = \{1, ..., T\}$ time horizon.
- r_{jt} is the return of the asset j at time t (ie is the realization of the random variable R_j during period t).
- $\bar{r_j} = (\Sigma_{t=1}^{T} r_{jt})/T$.
- x_j money invested in asset j.

Now, the investors problem will be restated with the new risk measure in the following way:

Minimize:

$$Min_x(E[|R_p - r_p|]), \tag{2.14}$$

subject to

$$\Sigma_{j=1}^{n} x_j r_j \geq \rho, \tag{2.15}$$

and

$$\Sigma_{j=1}^{n} x_j = 1. \tag{2.16}$$

When this model is implemented, $\sum_{t=1}^{T} r_{jt}/T$ is used as an estimator of $E(R_j)$. Therefore, the problem that we are now facing is the following:

Minimize:

$$Min_x(\sum_{t=1}^{T} |\sum_{j=1}^{n} x_j(r_{jt} - \bar{r_j})|)\frac{1}{T}, \tag{2.17}$$

subject to

$$\sum_{j=1}^{n} x_j r_j \geq \rho, \tag{2.18}$$

and

$$\sum_{j=1}^{n} x_j = 1, \tag{2.19}$$

where $\bar{r_j} = \sum_{t=1}^{T} r_{jt}/T$.

Now, with only a few adjustments, we can state the problem in a way such that is solvable by linear programming process. Let us denote: $a_{jt} = r_j t - \bar{r_j}$. Then, we have: Minimize:

$$Min_x(\sum_{t=1}^{T} |\sum_{j=1}^{n} x_j a_{jt}|\frac{1}{T}) \tag{2.20}$$

subject to

$$\begin{aligned} \sum_{j=1}^{n} x_j r_j &\geq \rho \\ \sum_{j=1}^{n} x_j &= 1 \end{aligned} \tag{2.21}$$

Which is equivalent to the following: Minimize:

$$Min_{x,y}(\sum_{t=1}^{T} y_t \frac{1}{T}) \tag{2.22}$$

subject to

$$\begin{aligned} y_t + \sum_{j=1}^{n} x_j a_{jt} &\geq 0, \qquad t = 1, ..., T, \\ y_t - \sum_{j=1}^{n} x_j a_{jt} &\geq 0, \qquad t = 1, ..., T, \\ \sum_{j=1}^{n} x_j r_j &\geq \rho, \\ \sum_{j=1}^{n} x_j &= 1. \end{aligned} \tag{2.23}$$

Note that, again, this formulation lacks an analytical solution. However, unlike the Markowitz problem, we can solve this by linear programming process.

Some of the differences between this model and the Markowitz model, already enumerated in the paper of the first appearance of MAD, are the following [**KY**]:

- It is not necessary to operate with a covariance matrix. This also facilitates the update of the model every period. This is important because when we invest, we should consider that the context of the market may vary, and therefore, we should balance the portfolio every now and then to confront those changes. Having to calculate the covariance matrix in environments constantly with so many data can be a problem.
- The linear problem is much easy to solve. Moreover, notice that the number of constraints here is constant (to be precise, it is 2T + 2) regardless the number of stocks. Therefore, it is feasible to solve this problem with thousands of stocks in an acceptable amount of time.
- The optimal solution of the MAD problem contains at most 2T+2 positive investments. Therefore, our portfolio will never have more than 2T+2 assets, no matter how big the universe of stocks is (ie n). This allows investors to use T as a control variable. As already mentioned, there are other constraints in real life that are not discussed in this paper, that have to be taken into account when doing real investments, and one of those might be the number of different stocks in which someone is willing to invest.

Notice that, again, with an analogous proof that we will omit, the One and Two fund theorems hold as true in this model (but with a graph where the the x-axis represent the value of the MAD instead of the standard deviation of every asset and portfolio).

2.3.1 Adding constraints

It is known that this models presented here are just a simplification of the real life. In fact, the constraints given are not the only ones that traders and investors face when making a decisions. There are some issues like legal problems (sometimes the exposure to certain assets or the level of leverage and debt is not allowed), the indivisibility of one unit of stocks (in general, you can not buy, for example, 10^{-7} stocks) among many others. At the end of the day, the important think is how this constraints will affect the feasibility of the achievement of a solution in a reasonable time when we are dealing with big data.

In the following lines, we will present 2 very simple constraints that are in fact real in many cases. Then we will evaluate how this affects the computational complexity (all of the following and further deeper analysis is developed in **[CST]**).

- Cardinality constraint: no more than K different assets should be held in the portfolio.

- Quantity constraint: the quantity x_i of each asset that is included in the portfolio has to be in a given interval $[l_i, u_i]$ (also called "buy-in threshold").

If we add just this reasonable restrictions (cardinality and quantity constraints), suddenly it turns out our problem is much more complicated, computationally speaking. The problem using the MAD model would look like the following:

Minimize:

$$Min_{x,y}(\sum_{t=1}^{T} y_t \frac{1}{T}), \tag{2.24}$$

subject to

$$\begin{aligned}
&y_t + \sum_{j=1}^{n} x_j a_{jt} \geq 0, \qquad t = 1, ..., T, \\
&y_t - \sum_{j=1}^{n} x_j a_{jt} \geq 0, \qquad t = 1, ..., T, \\
&\sum_{j=1}^{n} x_j r_j \geq \rho, \\
&\sum_{j=1}^{n} x_j = 1, \\
&\sum_{j=1}^{n} y_j \leq K, \\
&l_i y_i \leq x_j \leq u_i y_j, \forall j \in I \\
&y_j \in \{0, 1\}
\end{aligned} \tag{2.25}$$

and

$$d_j \geq 0. \tag{2.26}$$

This problem has to be solved by Mixed Integer Linear Programing [**SEW**], and therefore, it falls into the class of NP-hard problems. Again, we face the impossibility of using this model when dealing with big data because of the complexity when more constraints are added.

2.4 Multiple Betas model and another extensions

The Multiple Beta models is a family of simple extensions of the CAPM model. Still, the accuracy is much higher. However, that does not mean we have the computational problems that we found in the Markowitz model. In those models, we are doing again a linear regression, but instead of only doing it with 1 factor (the market portfolio R_m), we will define more factors. There are plenty of variations and versions of this idea. Some of the most popular are the Fama-French 3 factor model (which we will describe below) or the Carhart four-factor model. Given that all are fairly similar, we are going to focus only on the first one (FF 3 factor).

2.4.1 Fama - French 3 factor model

This model was created by two professors from the University of Chicago: Eugene Fama and Kenneth French. This was one of the most popular models when dealing with big data in the stock market, because of its simplicity and accuracy. Now a days, the models used are small variations of it, usually adding new factors like momentum.

The 3 factors that are used in this model are the following

1. Market: this factor represents the same as it did in the CAPM model. It is the return of the market portfolio.
2. SMB (**S**mall -market capitalization - **M**inus **B**ig)
3. HML (**H**igh -book to market ratio - **M**inus **L**ow)

Now we are going to give a small explanation of what are they supposed to represent and how they are estimated. In the following lines, we will introduce new concepts like market capitalizations and book to market ratio.

- **Market capitalization** := number of shares of the company $\times$ price of every share.
- **Book to market ratio** := $\frac{(A-B)}{C}$, where
    - A = sum of the value of all the assets of the company,
    - B = sum of the value of all liabilities,
    - C = market capitalization.

Note that $(A - B)$ = value of the company according to the accountancy.

When people want to differentiate the stocks depending on their market capitalization, they usually refer to them as low cap, mid cap and small cap. When people want to differentiate stocks depending on their book to market ratio, they usually use the concepts of growth stock (stock with low BTM ratio. BTM$<$ 1) and value stock(stock with high BTM ratio . BTM$\geq$1).

Now we are going to explain a bit the reason of the selection of the new two factors (SMB, HML) for the regression and ways of estimating them:

SMB (small -market capitalization - minus big): Empirically, it has been observed that the returns of stocks of companies with small market capitalization are higher, maintaining other criteria constant. The literature has found a few reasons justifying this, like the less risk of big companies to fail in market crashes, or the overvaluation of famous big companies (investors like to buy shares of Apple or Google just because they are famous). One simple and naive driver of this factor would be a portfolio that, after reordering all shares in the market depending on their market capitalization, longs the top 10% of those companies and shorts the bottom 10%.

HML (high -book to market ratio - minus low): Again, empirically, it has been seen that those companies with high book to market ratio perform better than others. Some reasons that have been found as origin of this are again the overvaluation of famous companies (usually, companies in the IT sector have most of their valuation based on intangible). Moreover, in general, the performance of growth stocks, when is positive, it outperforms value stocks, which perform consistently better in the long run, but with

lower and more stabilized returns. This causes the so called recency effect (people buy stocks based on recent returns, ignoring the long run), causing again an overvaluation of the growth companies and an undervaluation of the value companies. One simple and naive driver of this factor would be a portfolio that, after reordering all shares in the market depending on their book to market ratio, longs the top 10% of those companies and shorts the bottom 10%.

The notation and assumptions in this model are the following:

- We have a universe of assets $\{R_i\}_{i \in I}$ where $I = \{1, ..., n\}$ plus 4 particular assets called the "risk free asset" R_f (this represents an asset like bonds, that have a fixed return and are considered free of risk in general), the "market portfolio" R_m, the "growth portfolio" R_g , and the "value portfolio" R_v.
- R_k behaves as a normal distribution s.t. $\sigma_k := var(R_k)$ and $r_k := E(R_k)$ for $k \in \{v, g, m, f\}$.
- For every asset $R_i \exists \alpha_i, \beta_i^m, \beta_i^v, \beta_i^g \in \mathbb{R}$ s.t. the following equality holds:

$$R_i - R_f = \alpha_i + \beta_i^m (R_m - R_f) + \beta_i^v R_v + \beta_i^g R_g + \epsilon_i,$$

  where ϵ_i is a normal distribution with $E(\epsilon_i) = 0$ and a certain variance σ_i(this ϵ_i adds to the model what we call "the idiosyncratic risk").
- $\forall i, j (i \neq j)$ then $covariance(\epsilon_i, \epsilon_j) = 0$.

Trivially, all the important results and conclusions explained in the CAPM model are true here. The problem of the investor is still a linear programing solvable (therefore, the algorithm to solve it has a polynomial time complexity) , but now, only by adding this 2 new factors, the accuracy of the model when used in real life is much higher when dealing with diversified portfolios and funds. The One and Two fund theorems also hold true in this model.

2.4.2 Proposed model: Clustered Betas model

During the study of the models with noticed the following: if the assets truly behave as certain distributions, then it shouldn't be necessary to check information from outside the historical returns. Theoretically, from the historical itself, one should be able to determine which is the distribution that the stock is following. According to this idea, it should not be necessary to check for information like the number of share, the value of the company in the accountancy or the index S&P500 to understand the distribution of any given stock.

Moreover, in the multiple beta model, as in CAPM model, the model proposes the factors to use before starting the regression. In our proposal, we will generate "factors portfolios" with the information of the historical itself, and nothing else. A factor portfolio will just be an hypothetical portfolio of assets that share a characteristic that cause a

high correlation among them (e.g. it could be their geographical situation or the fact that they are operating in the same sector). Still, the only criteria to create those factors is that they are highly correlated and we do not need to check if they indeed have something in common. The curious thing about our proposal is that we might identify factors that otherwise we would have never identified, that are impossible to see unless one checks for the historical returns of the assets, or that even after being identified we do not understand. The factors also will be particularly specific for every market. Moreover, this way of creating the factors can be used in markets outside the stock market (like commodities), in which other multiple beta factors like the Fama French do not fit because of a selection of the factors specifically for shares of companies.

We propose the following algorithm to put these ideas in practice:

1. First of all, we have to define a number $c \in [0,1]$ that represents the absolute value of what we will consider a high correlation, and k the number of factors we want to use. In our implementation, we take as $c =$ *percentile 97.5 of all the correlations among all assets* , and $k = 3$, imitating FF3factor. Notice that this is just one approach to select what we consider a fair number of factors and a high correlation, but different criteria can be use to choose c and k.

2. Define m = 0

3. For all $i \in I$ we set $S_i := \{i \in I : |\sigma_{ij}| \geq c\})$. This set represents a factor, so we have one factor for each stock. Now we have $\#I$ factors, which are too much to do a regression (we only need k). We will use only the *best* factor in the following step. Notice this set contains *similar* stocks, in the sense that all elements of any set S_i are highly correlated with stock i.

4. Take the set S_i with higher number of elements. This will be a factor. Now we will create the following portfolio as an estimator of S_i: invest +1$ in each stock $j \in J$ s.t. $\sigma_{ij} > 0$ and shortsell -1$ in each stock $j \in J$ s.t. $\sigma_{ij} < 0$. Call this portfolio F_m.

   If you have done this k times (ie $m = k - 1$), continue to step 5. Otherwise, we rename $I := I \backslash S_i$, $m = m + 1$ and return to step 3.

5. Do a OLS linear regression with the $k - 1$ "factor portfolios" $F_1, ..., F_{k-1}$ and the "market portfolio" to find $\alpha, \beta_1, ..., \beta_{k-1}, \beta_M$.

Chapter 3

IMPLEMENTATION OF THE MODELS IN PYTHON

In the following lines, we are going to explain some experimental comparisons that are done with the models, in order to have a more practical view of the differences between them and a reflex of the problems and results that were predicted in the theory previously.

3.1 MAD vs Markowitz

In the following lines, we are going the optimal portfolios calculated by 2 different programs for very different contexts. As input, we will give the historical of stocks of the US Market. All of them will be part of the S&P100, but we will check the results in different dates and with different quantities of stocks.

The data was extracted from the webpage www.quantopian.com, which has proved to be an extremely useful tool during the realization of this thesis to find code implementing strategies, historical data of the US stocks market and discussions about finance topics.

Before starting the analysis, I would like to refer to results obtained in previous research by other authors([**BW**], [**KY**]).

In the original paper where the MAD model was presented [**KY**], they compared its results with 224 included in NIKKEI 225 index. In their paper, they showed that for that specific data, the portfolios obtained for one or the other method were quite similar. In particular, the difference of the portfolios in the standar deviation was at most a 10 % of the value of ρ (so apparently the accuracy of the replace depends on the demanded minimum average return).

In the studies of [**BW**], they compared 30 portfolios consisting of five stocks and a six-month bond by randomly selecting the stocks from the S&P 500. Roughly, their paper

conclude that, for a small quantity of stocks, the two models (MAD and Markowitz) give similar results.

Case with 5 stocks

First of all, we should know some information about the daily returns of the S&P100 to have a bit of background. For this reason, I introduce this table with the average daily return of the different percentiles of the stocks from the S&P100:

percentile 10	-1.09%
percentile 20	-0.62%
percentile 30	-0.35%
percentile 40	-0.13%
percentile 50	0.05%
percentile 60	0.24%
percentile 70	0.46%
percentile 80	0.76%
percentile 90	1.26%

Therefore, now we know that 50% of the stocks produce at least an average daily return of 0.05%, and that only the top 10% of the stocks have an average daily return higher than 1.26%.

Some other interesting facts are the following: the average daily return from all the stocks is 0.06%, the minimum daily return was -16%, the maximum was +11% and the variance was 1.15%.

The following results have been obtained using the closing price of the daily returns of the following stocks:
"AAPL", Apple Inc., Tech company
"AMZN", Amazon.com, Inc., electronic commerce
"GOOG", Alphabet Inc Class C (GOOGLE), Tech company
"GS", Goldman Sachs Group Inc, financial services
"JPM", JPMorgan Chase & Co., financial services

start date="2016-07-01" - end date="2017-08-01"

In the image 3.1 we can see the blue dots representing the 5 stocks. In yellow color the efficient frontier given by the MV model, and in red the efficient frontier given by the MAD model. Obviously, the frontier given by the MV model is the one more to the left. It looks like they are really close in all tested values.

We have calculated the optimal portfolios of both MAD and Markowitz using the data between "2016-07-01" and "2017-08-01" with different minimum expected average return requirements (ρ). After that, we have checked the daily performance of those 2 portfolios

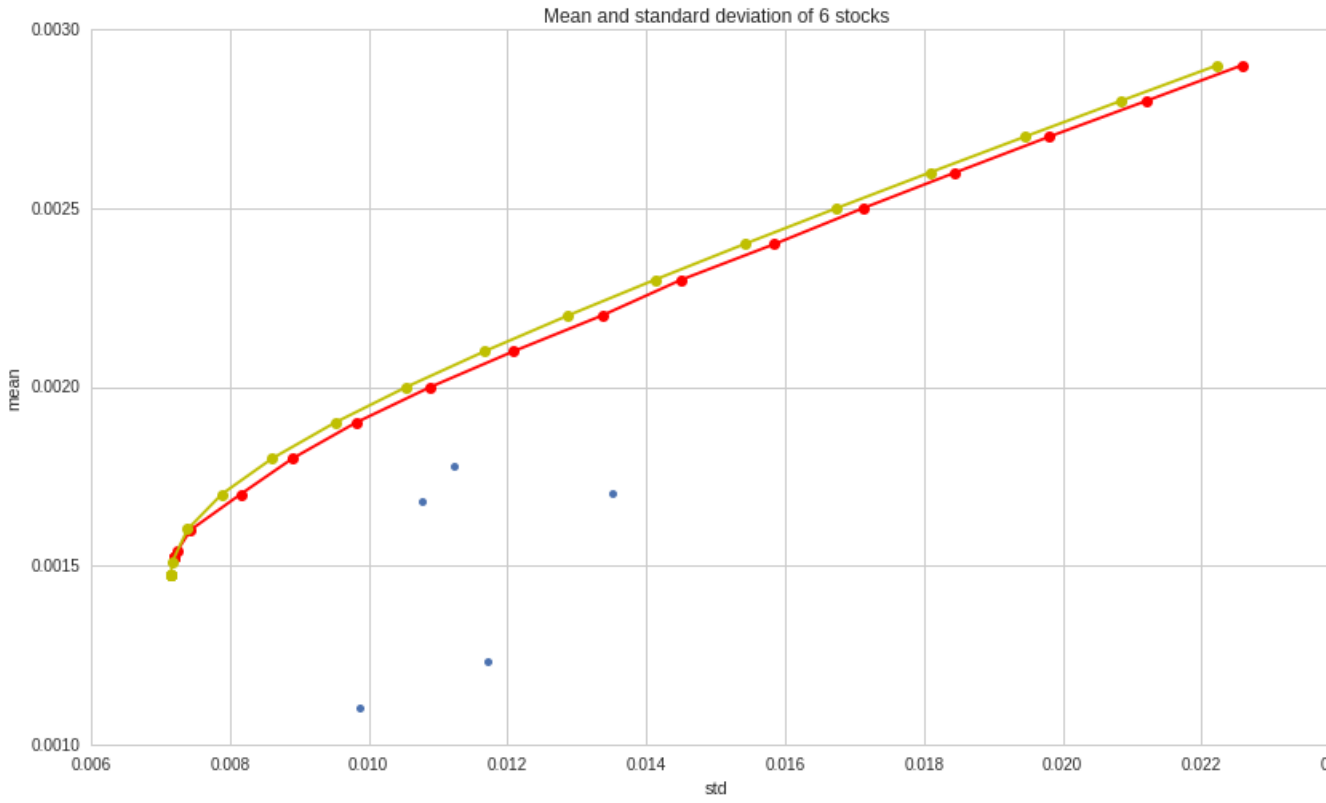

Figure 3.1: Graph of our implementation of the models in python with 5 different stocks

between the dates "2017-03-01" and "2017-08-01" for every ρ. With that data, we have done a paired t-test, checking for the following test of hypothesis testing:

$$H_0 : R_{MAD} - R_{MV} = 0$$
$$H_1 : R_{MAD} - R_{MV} \neq 0.$$

ρ	p value
0.02%	0.3949
0.04%	0.3949
0.06%	0.3949
0.08%	0.3949
0.1%	0.3949
0.2%	0.5662
0.4%	0.8785
0.6%	0.7559
0.8%	0.6999
1%	0.8207
2%	0.6434
4%	0.6016

As we can see, in all the checked ρ, we do not reject the null hypothesis at the 10% significance level. Therefore, we conclude that we will consider that the returns of the portfolios given by the 2 different models have similar daily returns.

One curious thing the reader might have notice in the table is the similar p-values when the minimum required average return of the portfolio is less than 0.1%. The reason for this is that, indeed, at all those low values the portfolio is the same. In all cases, the portfolio will have around 0.1% return, independently of the minimum required, because we are requiring a ρ lower than any value of the efficient frontier. This means that if we wanted a portfolio of exactly an average return ρ (this means, a return lower than the minimum of the efficient frontier), we would have to increase our risk! This is clearly worse off. Therefore, this explains the similar values.

We wanted to do another test to compare the weights given to all stocks. Theoretically, if both models are close, the weight to invest in each stock should be he same in both models. Therefore, we do the following test of hypothesis to check it:

$$H0 : w_i(MAD)/w_i(MV) = 1$$
$$H1 : w_i(MAD)/w_i(MV) \neq 1$$

Using all the weights from all the portfolios we have calculated so far, the result we obtained was the following:

$$t = -1.06662407899$$

$$p - value = 1 - 0.713858377225 = 0.29$$

Therefore, we can not reject the null hypothesis at a 10% level of signification, as we expected.

Case with 100 stocks

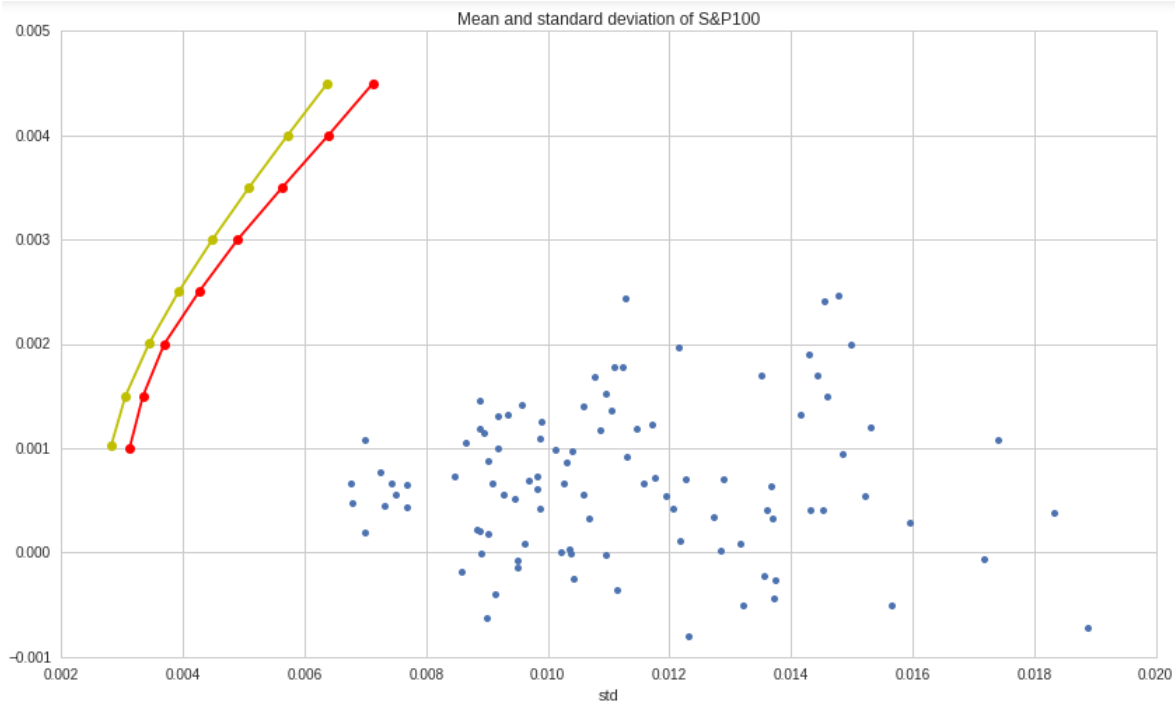

Figure 3.2: Graph of our implementation of the model in python(the S&P100)

The following results have been obtained using the closing price of the daily returns of the stocks of the S&P100: start date="2016-07-01" - end date="2017-08-01"

First of all, we are going to check the graph of the stocks and the 2 efficient frontiers. Again, in the image 3.2 we can see the blue dots representing the different stocks of the S&P100, in color yellow the efficient frontier given by the MV model, and in color red the efficient frontier given by the MAD model. As before, the frontier given by the MV model is the one more to the left. However, on the first sight, it looks like they are really close, and only for high values of minimum required return they start to diverge.

As we can see, comparing this graph with the previous one, the more stocks we have, the more far away from the stocks is the frontier. This is because with more stocks, we can diversify better the idiosyncratic risk and obtain much better combination of stocks in the

portfolios, increasing considerably the performance.

We now repeat the hypothesis test done previously in the case of 5 stocks. The table that we obtain is the following:

ρ	p value
0.02%	0.4761
0.04%	0.4761
0.06%	0.4761
0.08%	0.4761
0.1%	0.4761
0.2%	0.7374
0.4%	0.6076
0.6%	0.5717
0.8%	0.5879
1%	0.7265
2%	0.6038
4%	0.6173

As we can see, the results do not change from the ones obtained in the case with only 5 stocks: we do not reject the null hypothesis at a 10% significance level. Moreover, we can see also here the same behavior of portfolios with low required minimum average return.

Apart from that, we have also repeated the hypothesis test of the weights, obtaining the following results:

$$t = -1.21332181178$$

$$p - value = 1 - 0.774993193196 = 0.225$$

Therefore, we again conclude that, as we can not reject the null hypothesis, we will consider that both models give similar weights.

From the tables above we learn a few things: as predicted, the two models give similar portfolios as the solution of the investor problem in terms of return. So far, it doesn't look like the minimum required average or the number of portfolios affects this. Therefore, in concordance with the results of [**KY**], unless other restrictions are added that make the model MAD computationally complex, MAD appears to be a good model to replace MV when dealing with big data.

However, there is one important thing that the reader should note. In all the hypothesis test that we have done, we have not rejected the null hypothesis, which was the default assumption. Therefore, further comparisons need to be done to prove with strength the similarity of the results between MAD and Markowitz.

3.2 Test of our proposed model (cluster betas model)

Now we will test the performance of our proposed model compared with the classical CAPM and the 3 Factor model. It looks obvious that, if we add more factors in the OLS linear regression, the R-square will increase. However, what we have to check here is if the adjusted R-square increases and the factors are statistically significant (so the factors indeed are there and drive the value of the stock).

In the following table, we compare the adjusted R-squared of four models in all stocks: CAPM, Fama French 4 factor model, Clustered Beta model (0.7) and Clustered Beta model(0.7) with only *clustered factors* and without the Market portfolio as one factor (we wanted to check how good would our idea perform without the "help" of the market factor).

The number that is in between parenthesis near the Clustered Beta models is the minimum correlation used to search for the factors (ie we are working with $c = 0.7$ and $k = 3$).

FF - 4 factor model is an improved version of the Fama French 3 factor that also includes "momentum" as a factor (momentum can be represented by a portfolio in which you long the top 10% stocks that better performed in the last year -except the previous month- and short the worse 10%, ie if it is 1st January 2018, you only need to search for the stocks that best performed between 1st January 2017 and 31st November 2017).

percentile	CAPM	Clust.βs-MKT(0.7)	C.βs+MKT (0.7)	FF4factor
0	0%	-0.0346%	0 %	1.0856%
10	7.3557%	04.6730%	9.7990 %	0.8871%
20	9.5166%	7.2024%	14.8273 %	1.6083%
30	12.4172%	10.0281%	18.2455 %	2.8159%
40	15.6522%	12.2932%	21.0644 %	4.9233%
50	17.9415%	15.6074%	23.5088 %	6.0985%
60	20.5795%	17.3194%	25.9767 %	7.7615%
70	23.2733%	23.3306%	33.8398 %	11.9194%
80	27.9085%	35.3876 %	40.83782 %	17.3301%
90	33.8735%	78.8648%	72.8515%	24.2938%
100	48.4141%	88.5445%	87.0452%	34.3134%

This results look quite good in comparison with all other models. As we can see from the table, the clustered beta model has an adjusted R-squared of over 23.5088% for half of the stocks of the S&P100. However, here we are only comparing the adjusted R-squared using stocks. In future research, our proposed model should be compared using entire funds, where it is known that Fama French 4 factor usually achieves an adjusted R-squared of 90%.

Apart from this, we have checked for each stock individually the p-value of each of the factors of the Clustered Beta model. In the large majority of the cases, the factors were

significant at a 10% level. Therefore, it looks like the factors are also statistically significant. However, we are aware that performing a t-test for each factor for each stock is not the best way to check for the statistical significance of the factors. Further results should be shown with better tests in posterior research to completely assure this.

Chapter 4

CONCLUSIONS, NEXT STEPS AND CODE

4.1 Conclusions

In this thesis, we have seen a quick review of the most important models in portfolio theory, focusing on the computational issue that some of them face when dealing with a large amount of stocks. Also, we have proposed a new model to approach the investors problem. During chapter 3, we have tested and compare the models using Python, and we have found some interesting results.

First, the comparisons between the MAD model and the MV model looks as we expected. Answering the question of [**BW**], it looks that the results are still maintained when dealing with a large number of stocks. Moreover, we have observed that depending on how extreme are the values that we require as minimum return, the portfolios are more or less similar. Despite that, the returns and variances of the overall portfolio are close in all cases in the graphs of the efficient frontiers. As next steps we would want to check if our results are maintained when we add restrictions to the feasible portfolio.

Secondly, I consider the idea of the proposed Clustered Betas model to have potential. As already stated, the model is still very naive, but has given promising results. It should be tested in more scenarios (like compare it with FF3factor using funds instead of individual stocks) and in different markets (e.g. commodities), to confirm that the model works as predicted.

During the realization of this thesis I did some research regarding clustered theory (theory consistent dividing a set of points into different classes, in which all the points of a class share an idea of "similitude") in order to find a more complicated way of clustering the market and obtaining the potential factors. However, mi initial trials using this kind of theory were unsuccessful when dealing with big data because of the high number of different arrangements that are possible (if we have n=1000 points -stocks- and we want

to divide them into 3 different classes - factors -, the numbers of ways to do this is greater than the number of particles in the universe). Therefore, most of the times, the algorithms that were tried to be implemented to calculate the factors ended up dealing with a NP-hard problem. Still, if a concept of distance based on the covariance is defined between the stocks, there might be algorithms in Cluster theory to implement a better version of the clustered betas model that do not require dealing with NP-hardness. Another possibility would be using of the covariance matrix in **spectral clustering** algorithms [**IS**], replacing the affinity matrix or the similarity matrix.

4.2 Code

The following code was used in the platform www.quantopian.com. Part of the code is taken from previous studies of the Markowtiz model and MAD model (see [**SEW**] and [**C**]).

This first code was used to compare the MAD and MV model (case with 100 stocks). Changing just the names of the stock in the return we can use again the program to check for different quantities of data.

```
from scipy.optimize import minimize
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import cvxopt as opt
from cvxopt import blas, solvers

# Computes the weights for the portfolio with the smallest Mean Absolute Deviation.
def MI_minimum_MAD_portfolio(returns, rho):
    avg_ret = np.mean(returns, axis=0)

    def _sum_check(x):
        return sum(x) - 1 #this function is used later for the constrain (\Sum x_i = 1)

    #the following is the inequality contrain
    def _min_return(x):

        a = np.dot(x, avg_ret)
        return (a- rho) #rho is the minimum desired return

    # Returns the Mean Absolute Deviation for the current iteration of weights
    def _mad(x, returns):
        return (returns - returns.mean()).dot(x).abs().mean() #this is MAD(x)
```

```
    num_assets = len(returns.columns)
    guess = np.ones(num_assets)  #This is a first guess.
    #The portfolio that invests 1 unit in everything

     #this are the constraints
    cons = ({'type':'eq', 'fun': _sum_check},  {'type': 'ineq', 'fun': _min_return})
    min_mad_results = minimize(_mad, guess, args=returns, constraints=cons)

    return pd.Series(index=returns.columns, data=min_mad_results.x)


def MI_minimum_MV_portfolio(returns, r_min):
    n = len(returns.T)
    returns = np.asmatrix(returns)


    # Convert to cvxopt matrices
    covs = opt.matrix(np.cov(returns.T))
    avg_ret = opt.matrix(np.mean(returns.T, axis=1))


    P = covs
    # x = variable(n)
    q = opt.matrix(np.zeros((n, 1)), tc='d')
    # inequality constraints Gx <= h
    # captures the constraints (avg_ret'x >= r_min) and (x >= 0)
    G = opt.matrix(np.transpose(np.array(-avg_ret)))
    h = opt.matrix(np.concatenate(np.ones((1,1))*(-r_min)))
    # equality constraint Ax = b; captures the constraint sum(x) == 1
    A = opt.matrix(1.0, (1,n))
    b = opt.matrix(1.0)
    sol = solvers.qp(P, q, G, h, A, b)['x']
    return np.asarray(sol)


returns = (get_pricing(["AAPL","ABBV","ABT","ACN" ,"AGN" ,"AIG" ,"ALL" ,"AMGN" ,
"AMZN" ,"AXP","BA","BAC","BIIB","BK","BLK","BMY","BRK.B",
"C","CAT","CELG","CHTR","CL","CMCSA","COF",
"COP","COST","CSCO","CVS","CVX","DHR","DIS","DUK","EMR","EXC","F",
"FB","FDX","FOX","FOXA",
"GD","GE","GILD","GM","GOOG","GS","HAL",
"HD","HON","IBM","INTC","JNJ","JPM","KHC","KMI","KO",
"LLY","LMT","LOW","MA","MCD","MDLZ","MDT","MET","MMM",
"MO","MON","MRK","MS","MSFT","NEE","NKE","ORCL","OXY","PCLN",
```

```
"PEP","PFE","PG","PM","PYPL","QCOM","RTN","SBUX","SLB","SO","SPG",
"T","TGT","TWX","TXN","UNH","UNP","UPS","USB","UTX","V","VZ","WBA","WFC","WMT","XOM"],
                            fields="close_price",
                            start_date="2017-01-01",
                            end_date="2017-03-01")).pct_change().dropna()
print  np.amax(np.array(returns)) , np.amin(np.array(returns))
print np.mean(np.array(returns)), np.sqrt(np.var(np.array(returns)))
for i in range(0, 100, 10):
        print "percentile ", i, "= ", np.percentile(np.array(returns), i)
rhos = [0.001, 0.002, 0.003, 0.004, 0.005, 0.006, 0.007, 0.008, 0.009, 0.1, 0.2, 0.3, 0.4]

for rho in rhos:
    weightsMAD = MI_minimum_MAD_portfolio(returns, rho)
    weightsMV = MI_minimum_MV_portfolio(returns, rho)
    Compare_MV_MAD=[]

    for security, weight in zip(returns, weightsMV):
        Compare_MV_MAD.append([security, weight[0], weightsMAD[security] ])

    from scipy import stats
    A =  np.array(Compare_MV_MAD).T
    X = A[1]
    Y = A[2]
    bet, alph, r_value, p_value, std_err = stats.linregress(X, Y)
    print "rho = ", rho
    print " alfa, beta, r2 =", alph, bet, r_value**2

rhos = np.arange(0.001, 0.005, 0.0005)
'''covs = np.cov(returnsM.T)
avg_ret =np.array(np.mean(returnsM.T, axis=1).T)'''
returnsM = np.asmatrix(returns)
covs = np.cov(returnsM.T)
avg_ret =np.array(np.mean(returnsM.T, axis=1).T)
MAD_ret_risk = [[], []]
MV_ret_risk = [[], []]
stds = np.diagonal(np.cov(returns.T))
for rho in rhos:
    weightsMAD = MI_minimum_MAD_portfolio(returns, rho)
    weightsMV = MI_minimum_MV_portfolio(returns, rho)
    Compare_MV_MAD=[]


    for security, weight in zip(returns, weightsMV):
        Compare_MV_MAD.append([security, [weight[0]], [weightsMAD[security] ]])
        #first element returns
```

```
    A =  np.array(Compare_MV_MAD).T
    MV = A[1]
    MAD = A[2]
    MAD = MAD.reshape((100, 1))

    #first element return
    rMAD =0
    rMV = 0
    for i in range(len(returns.T)):
        rMAD = rMAD + avg_ret[0][i]*MAD[i][0][0]
        rMV = rMV + avg_ret[0][i]*MV[i][0]
    MAD_ret_risk[0].append(rMAD)
    MV_ret_risk[0].append(rMV)

    #second element risk
    varMV=0
    aux = np.zeros(len(returns.T))
    for j in range(len(returns.T)):
            for k in range(len(returns.T)):
                aux[j] = aux[j] + covs[j][k]*MV[k][0]
    for i in range(len(returns.T)):
        varMV = varMV + MV[i][0]*aux[i]
    MV_ret_risk[1].append(np.sqrt(varMV))

    varMAD=0
    aux = np.zeros(len(returns.T))
    for j in range(len(returns.T)):
            for k in range(len(returns.T)):
                aux[j] = aux[j] + covs[j][k]*MAD[k][0][0]
    for i in range(len(returns.T)):
        varMAD = varMAD + MAD[i][0][0]*aux[i]
    MAD_ret_risk[1].append(np.sqrt(varMAD))

covs = opt.matrix(np.cov(returns.T))
avg_ret = opt.matrix(np.mean(returns.T, axis=1))
stds = np.sqrt(np.diagonal(np.cov(returns.T)))

import matplotlib.pyplot as plt

plt.plot(stds, avg_ret, 'o', markersize=5)
plt.xlabel('std')
plt.ylabel('mean')
plt.title('Mean and standard deviation of S&P100');
plt.plot(MAD_ret_risk[1], MAD_ret_risk[0], 'r-o');
plt.plot(MV_ret_risk[1], MV_ret_risk[0], 'y-o');
```

This second code was used to compare our proposed model with previous ones like CAPM (case with 100 stocks). Again, changing just the names of the stock in the return we can use again the program to check for different quantities of data. Part of the code was taken from [**NMGW**].

```
import numpy as np
import pandas as pd
import statsmodels.api as sm
# If the observations are in a dataframe,
#you can use statsmodels.formulas.api
#to do the regression instead
from statsmodels import regression
import matplotlib.pyplot as plt
from scipy import stats


def max_assoc_from_list(I,n, corr, min_corr):
#this function takes a list, returns element that
    #has more associated elements and the associated elements
n_assoc = np.zeros(n)
for i in I:
J = I[i+1:] #creas un conjunto J que indica el resto de I para comparar con i
for j in J:
if (abs(corr[i, j]) > min_corr):
n_assoc[i] += 1;
n_assoc[j] += 1;
# "factor" is the element with highest amount of associates
factor =  np.argmax(n_assoc)

#now we gather all the associated elements of "factor"
assoc = []
for i in I:
if (abs(corr[factor, i]) > min_corr):
assoc.append(i)

return factor, assoc


def reduce_list(factorRepres, I, min_corr, returns):
    corr = np.corrcoef(returns)

#I is taken and modified by reference, so there is no need to return anything
    for i in I:
     if (abs(corr[int (factorRepres), i ]) > min_corr):
```

```
     I.remove(i)


def generate_n_factors(returns, min_corr, nFactors):
    # WE CAN CHANGE MIN_CORR DEPENDING ON THE CONTEXT
    n = len(returns)
    corr = np.corrcoef(returns)
    #I is a list of all the stocks from where we generate a factor portfolio.
    #Each time a factor portfolio is generated,
    #I is actualized and deletes the stocks from that portfolio
    I = range(n)
    # "factor" is a subset of I that will tell us
    #which are the generators of the factors
    # "assoc" is a list of lists, and each element of
    #"assoc" will be all the associated elements of each factor
    factor = np.zeros(nFactors)
    assoc = []

    for k in range(nFactors):
        assoc.append([])
        factor[k], assoc[k] = max_assoc_from_list(I, n, corr, min_corr)
        #Now this eliminates from I all the elements that are already in a factor portfolio
        reduce_list(factor[k], I, min_corr, returns)

    return (factor, assoc)


def factors_history(returns,  min_corr, nFactors):
    (factor, assoc) =  generate_n_factors(returns,  min_corr, nFactors)
    corr = np.corrcoef(returns)
    Mreturns = returns.values

     #Now let's create the portfolios that will serve
     #as estimators of the factors:
    #we will create the portfolios in a really naive way.
    #With more time, we would have checked for better estimations
    # our portfolios will long 1 dolar of all assets with
    #possitive correlation and short 1 dollar of all assets
    #with negative correlation
    #the only information that we need is the history of the returns
    hist_fact =  np.zeros( nFactors* returns.shape[1] ).reshape((nFactors, returns.shape[1]))
    for i in range(nFactors):#nFactors = 3
        nAss = 0#number of associated to the factor.
        #Used later to divide in the average return
        for element in assoc[i]: # for each of the elements
```

```
        #associated with the i-th factor, buy or sell it
            nAss +=1
            if corr[element, int (factor[i]) ] > 0:
                #long, buy 1 stock
                hist_fact[i] += Mreturns[element]
            else:
                #shorting, sell 1 stock
                hist_fact[i] -= Mreturns[element]
        hist_fact /= nAss
        #notice the return of each of this porfolio should be the
        #signed sum of all the returns divided by the number of
        #stocks associated with that cluster

    return  hist_fact
start = "2016-07-01"
end = "2017-08-01"

returns = get_pricing(["AAPL","ABBV","ABT","ACN" ,"AGN" ,"AIG" ,"ALL" ,"AMGN" ,
"AMZN" ,"AXP" ,"BA" ,"BAC","BIIB","BK","BLK","BMY","BRK.B","C","CAT","CELG",
"CHTR","CL","CMCSA","COF","COP","COST","CSCO","CVS","CVX","DHR","DIS","DUK"
,"EMR","EXC","F","FB","FDX","FOXA","GD","GE","GILD","GM","GS","HAL","HD","HON",
"IBM","INTC","JNJ","JPM","KHC","KMI","KO","LLY","LMT","LOW","MA","MCD","MDLZ",
"MDT","MET","MMM","MO","MON","MRK","MS","MSFT","NEE","NKE","ORCL","OXY","PCLN",
"PEP","PFE","PG","PM","PYPL","QCOM","RTN","SBUX","SLB","SO","SPG","T","TGT",
"TWX","TXN","UNH","UNP","UPS","USB","UTX","V","VZ","WBA","WFC","WMT","XOM"],
                            fields="close_price",
                            start_date=start,
                            end_date=end).pct_change().dropna()
market = get_pricing('SPY', fields='price', start_date=start, end_date=end).pct_change().dropna
R_F = get_pricing('BIL', fields='price', start_date=start, end_date=end).pct_change().dropna()
EXMRKT = market - R_F

#Clustered beta + market factor
list_min_corr = [0.6, 0.65, 0.7,0.75, 0.8,0.85 ]
#for min_corr in list_min_corr:
nFactors = 2
min_corr = 0.8
hist_fact = factors_history(returns.T, min_corr, nFactors)
# in this variable we have 3 vectors with the historical prices of the


X = sm.add_constant( np.column_stack( (EXMRKT, hist_fact[0], hist_fact[1]) ) )
betas = range(returns.values[1].size)
# Run the model
for i in range(returns.values[1].size): #(returns.values[1].size) = 100 ,
```

```
#ie the number of stocks
    Y = (returns.T.values[i] - R_F)#historical of each stock
    betas[i] = regression.linear_model.OLS(Y, X).fit()
adj_R = np.zeros(len(betas))
for i in range(i):
    #print betas[i].summary()

    adj_R[i] = betas[i].rsquared_adj
print min_corr
for i in range(11):
    print i*10, np.percentile(adj_R, i*10)

#Clustered beta Model

list_min_corr = [0.6, 0.65, 0.7,0.75, 0.8,0.85 ]
#for min_corr in list_min_corr:
nFactors = 3
min_corr = 0.8
hist_fact = factors_history(returns.T, min_corr, nFactors)
# in this variable we have 3 vectors with the historical prices

X = sm.add_constant( np.column_stack( (hist_fact[0], hist_fact[1], hist_fact[2]) ) )
betas = range(returns.values[1].size)
# Run the model
for i in range(returns.values[1].size): #(returns.values[1].size) = 100 ,
#ie the number of stocks
    Y = returns.T.values[i] #historical of each stock
    betas[i] = regression.linear_model.OLS(Y, X).fit()
    #print len(X), len(Y)
for i in range(i):
    #print betas[i].summary()
    #to compare atributes, check
    #http://www.statsmodels.org/dev/generated/statsmodels.regression.linear_model.RegressionResu
    adj_R[i] = betas[i].rsquared_adj
print min_corr
for i in range(11):
    print i*10, np.percentile(adj_R, i*10)

#CAPM


X = sm.add_constant( EXMRKT )
betas = range(returns.values[1].size)
# Run the model
for i in range(returns.values[1].size): #(returns.values[1].size) = 100 ,
```

```
#ie the number of stocks
    Y = (returns.T.values[i] - R_F)#historical of each stock
    betas[i] = regression.linear_model.OLS(Y, X).fit()
adj_R = np.zeros(len(betas))
for i in range(i):
    #print betas[i].summary()

    adj_R[i] = betas[i].rsquared_adj
print "CAPM"
for i in range(11):
    print i*10, np.percentile(adj_R, i*10)
print "NOW NOT ADJUSTED R2"

for i in range(i):
    #print betas[i].summary()
    adj_R[i] = betas[i].rsquared
```

Bibliography

[1] [LU] David G.Luenberger: *Investment Science*. Oxford University Press, New York (1998).

[2] [MA] Harry M. Markowitz:*Portfolio Selection*. Journal of Finance, Volume 7, Number 1 (1952), pages 77-91.

[3] [MW] Beth Bower, Pamela Wentz: *Portfolio optimization: MAD vs Markowitz*. Rose - Hulman Undergraduate Mathematics Journal, Volume 6, Issue 2, article 3.

[4] [CST] Francesco Cesarone, Andrea Scozzari, Fabio Tardella: *Portfolio selection problems in practice: a comparison between linear and quadratic optimization models*. arXiv:1105.3594 [q-fin.PM] (July 2010).

[5] [KY] Hiroshi Konno, Hiroaki Yamazaki: *Mean-Absolute Deviation Portfolio Optimization Model and its applications to Tokio Stock Market*. Management Science. Volume 37, number 5 (May 1991), pages 519-531.

[6] [FF] Eugene F. Fama, Kenneth R. French: *The Capital Asset Pricing Model: Theory and Evidence*. Journal of Economic Perspectives. Volume 18, number 3 (Summer 2004), pages 25-46.

[7] [IS] Laura Igual, Santi Segui: *Introduction to Data Science: a python approach to concepts techniques and application*, Springer.ISBN 978-3-319-50017-1. Barcelona (2017).

[8] [LE] Jan van Leeuwen: *Handbook of Theoretical Computer Science. Vol. A, Algorithms and complexity*. Elsevier. ISBN 0262720140. OCLC 247934368. Amsterdam (1998).

[9] [SEW] Thomas Starke, David Edwards, Thomas Wiecki: *The Efficient Frontier: Markowitz Portfolio optimization in Python*. https://blog.quantopian.com/markowitz-portfolio-optimization-2/

[10] [C] James Christopher: *MAD Portfolio an alternative to Markowitz?*. https://www.quantopian.com/posts/mad-portfolio-an-alternative-to-markowitz

[11] [NMGW] Evgenia Nitishinskaya, Maxwell Margenot, Delaney Granizo-Mackenzie, Gilbert Wasserman: *Multiple Linear Regression*. https://www.quantopian.com/lectures/multiple-linear-regression

[12] [F] Craig W. French: *The Treynor Capital Asset Pricing Model*. Journal of Investment Management. Volume 1, number 2 (2003), pages60 - 72. SSRN 447580.

[13] [A] Glen Arnold. *Corporate financial management* (3. ed.). Harlow [u.a.]: Financial Times/Prentice Hall. (2005), page 354.

[14] [R] C. R. Rao: *Linear Statistical Inference and its Applications*. John Wiley and Sons, Inc.. New York (1965).

[15] [Kh] Leonid Khachiyan: *Mathematics of Operations Research archive*. Volume 5, Issue 1 (February 1980), pages iv-iv.