



**NOVA**

**IMS**

Information  
Management  
School

# MAAA

---

**Mestrado em Métodos Analíticos Avançados**  
Master Program in Advanced Analytics

## **PORTFOLIO OPTIMIZATION USING EVOLUTIONARY ALGORITHMS**

Alejandro Alvarez Etchegaray

Dissertation presented as the partial requirement for  
obtaining a Master's degree in Data Science and Advanced  
Analytics

**NOVA Information Management School**  
**Instituto Superior de Estatística e Gestão de Informação**  
Universidade Nova de Lisboa

**NOVA Information Management School**  
**Instituto Superior de Estatística e Gestão de Informação**  
Universidade Nova de Lisboa

PORTFOLIO OPTIMIZATION USING EVOLUTIONARY ALGORITHM

by

Alejandro Alvarez Etchegaray

Dissertation presented as the partial requirement for obtaining a Master's degree in Data Science and Advanced Analytics

**Advisor: Mauro Castelli**

March, 2021

# Abstract

Portfolio optimization is a widely studied field in modern finance. It involves finding the optimal balance between two contradictory objectives, the risk and the return. As the number of assets rises, the complexity in portfolios increases considerably, making it a computational challenge. This report explores the application of the Multi-Objective Evolutionary Algorithm based on Decomposition (MOEA/D) and Genetic Algorithm (GA) in the field of portfolio optimization.

MOEA/D and GA have proven to be effective at finding portfolios. However, it remains unclear how they perform when compared to traditional approaches used in finance. To achieve this, a framework for portfolio optimization is proposed, using MOEA/D, and GA separately as optimization algorithms and Capital Asset Pricing Model (CAPM) and Mean-Variance Model as methods to evaluate portfolios.

The proposed framework is able to produce weighted portfolios successfully. These generated portfolios were evaluated using a simulation with subsequent (unseen) prices of the assets included in the portfolio. The simulation was compared with well known portfolios in the same market and other market benchmarks (Security Market Line and Market Portfolio).

The results obtained in this investigation exceeded expectation by creating portfolios that perform better than the market. CAPM and Mean-Variance Model, although they fail to model all the variables that affect the stock market, provide a simple valuation for assets and portfolios. MOEA/D using Differential Evolution operators and the CAPM model produced the best portfolios in this research.

Work can still be done to accommodate more variables that can affect markets and portfolios, such as taxes, investment horizon and costs for transactions.

# Graphical Abstract

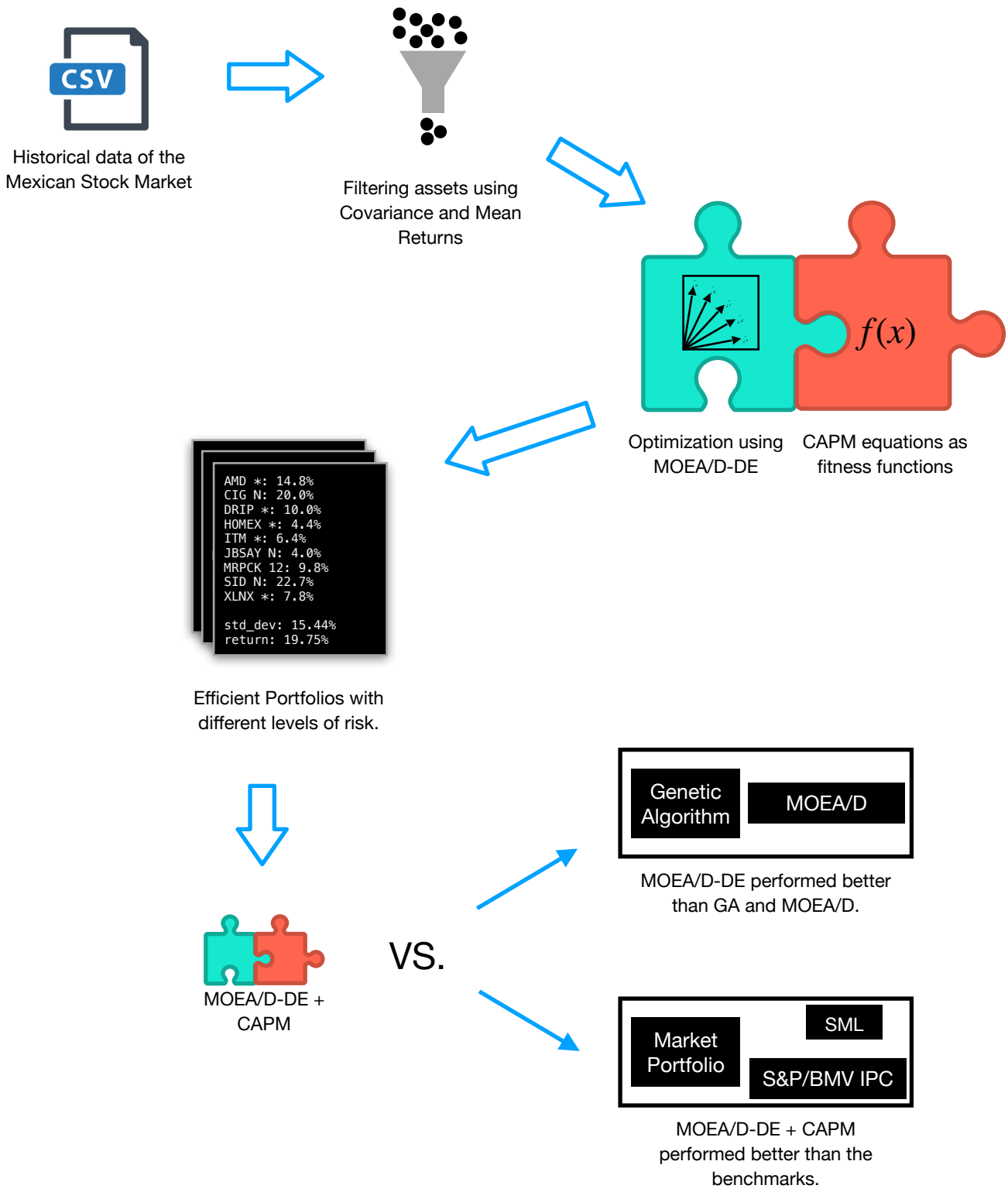


Figure 0. Graphical Abstract

# Table of Contents

1. Introduction	1
2. Theoretical Framework	4
2.1 Investment Theory	4
2.1.1 Mean-Variance Portfolios	4
2.1.2 Capital Asset Pricing Model	6
2.2 Traditional Methods for Portfolio Creation	8
2.2.1 Selecting Assets	8
2.2.2 Optimizing the Portfolio	9
2.3 Evolutionary Algorithms	9
2.3.1 Genetic Algorithm	11
2.3.2 Multi-Objective EA Based on Decomposition	13
2.3.3 Differential Evolution	17
2.4 Literature Review	19
3. Methodology	23
3.1 Tools and Methods	23
3.2 Implementation	24
3.2.1 Genetic Algorithm	25
3.2.2 MOEA/D	29
3.2.3 Stock Pre-Selection	31
4. Experimental Phase	33
4.1 Data	33
4.2 Metrics	34
4.3 Benchmarks	35
4.4 Procedure	36
4.4.1 Parameter Optimization	37
4.5 Results and Discussion	37
4.5.1 Optimization Comparison	38
4.5.2 Simulations and Portfolio Model Comparison	41
4.5.3 Comparison Against Benchmarks	45
4.5.4 Discussion	47
5. Conclusion and Future Work	49
6. Bibliography	50
Appendix 1: Systematic Literature Review Table	a
Appendix 2: Algorithm Configuration	c
Appendix 3: Parameter Optimization Results	e
Appendix 4: Results Summary	k
Appendix 5: Portfolios Found	m

# List of Figures

Figure 0. Graphical Abstract	VI
Figure 1. Efficient Frontier	6
Figure 2. Security Market Line	7
Figure 3. Evolutionary Algorithm	10
Figure 4. Genetic Algorithm	11
Figure 5. MOEA/D Decomposition	14
Figure 6. MOEA/D Neighborhoods	15
Figure 7. Methodology	22
Figure 8. Inputs and Outputs of the Proposed Solution	23
Figure 9. Initialization Methods Comparison	26
Figure 10. Effects of Normalization	29
Figure 11. Market Portfolio	32
Figure 12. Dataset Split	32
Figure 13. Mean-Variance Portfolios	36
Figure 14. MOEA/D vs. MOEA/D	37
Figure 15. CAPM Portfolios	38
Figure 16. Genetic Algorithm SML Simulation	29
Figure 17. Genetic Algorithm Portfolios' Time Series	40
Figure 18. MOEA/D SML Simulation	41
Figure 19. MOEA/D Portfolios' Time Series	41
Figure 20. MOEA/D-DE SML Simulation	42
Figure 21. MOEA/D-DE Portfolios' Time Series	43
Figure 22. All Portfolios in SML	44
Figure 23. All Portfolios vs. Market Portfolio	44
Figure 24. All Portfolios vs. Market Index	45

## List of Tables

Table 1: Guidelines for reading $\beta$ values	8
Table 2: Genetic Operators for GA	28
Table 3: Genetic Operators for MOEA/D	28

## List of Algorithms

Algorithm 1: MOEA/D[4]	17
Algorithm 2: Initialization of Population	27
Algorithm 3: Stock Pre-selection	30

## List of Abbreviations and Acronyms

<b>AI</b>	Artificial Intelligence
<b>MOEA/D</b>	Multi-objective Evolutionary Algorithm Based on Decomposition
<b>MOEA/D-DE</b>	Multi-objective Evolutionary Algorithm Based on Decomposition with Differential Evolution
<b>DE</b>	Differential Evolution
<b>GA</b>	Genetic Algorithm
<b>EA</b>	Evolutionary Algorithm
<b>MOP</b>	Multi-Objective Problem
<b>NSGA</b>	Non-dominated Sorting Genetic Algorithm
<b>NSGA-II</b>	Non-dominated Sorting Genetic Algorithm II
<b>CAPM</b>	Capital Asset Pricing Model
<b>MV</b>	Mean Variance Model
<b>SML</b>	Security Market Line
<b>EP</b>	External Population
<b>PSO</b>	Particle Swarm Optimization
<b>MOEO</b>	Multi-Objective Extremal Optimization
<b>MODE</b>	Multi-Objective Differential Evolution
<b>NDS</b>	Non-Dominated Sorting Algorithm
<b>MOCLPSO</b>	Multi-Objective Comprehensive Learning Particle Swarm Optimizer
<b>SPEA2</b>	Strength Pareto Evolutionary Algorithm 2
<b>PAES</b>	Pareto Archived Evolution Strategy
<b>MOPSO</b>	Multi-Objective Particle Swam Optimization
<b>PSFGA</b>	Parallel Single Front Genetic Algorithm
<b>M-CABC</b>	Multi-Objective Co-Variance based Artificial Bee Colony
<b>BIVA</b>	Bolsa Institucional de Valores
<b>BMV</b>	Bolsa Mexicana de Valores



# 1. Introduction

*Portfolio optimization has been around for more than 60 years.* It was first introduced in 1952 by Harry Markowitz[1] in his paper *Portfolio Selection*. This mathematical representation of portfolio optimization was awarded the Nobel Prize in Economics in 1990[24]. Specifically, *Portfolio optimization* is the process of selecting the best asset distribution from a set of assets (search space), to fulfill an objective[2]. This objective is typically a balance between the risk and the return of an investment. As the number of possible assets in the portfolio rises, the optimization of a portfolio becomes more complex, and exploring all possibilities becomes almost impossible[1][2].

Artificial intelligence (AI) methods have shown to be effective at finding suitable solutions to multi-objective problems. In past years, there have been multiple attempts to achieve optimal portfolio optimization using a variety of AI based methods. Using search methods like Simulated Annealing or Tabu Search has been common practice[25] and in recent years these multi-objective search type methods have gained popularity in the field[22]. Implementations using methods such Particle Swarm Optimization, Genetic Algorithm, Artificial Bee Colony have obtained success at portfolio optimization as described in the literature[15-23].

Multi-Objective Evolutionary Algorithms have previously been used for Portfolio Optimization[8][9][10]. This research proposes a framework to generate portfolios with Evolutionary Algorithms along with a new evaluation method that covers different aspects of finance and computer science to assess portfolios and algorithms.

Most of the research currently has been focused in Evolutionary Algorithms as a tool for optimization. However, it is unclear whether Evolutionary Algorithm optimized portfolios are profitable enough to be an attractive investment and if EAs can perform better than current traditional approaches when creating investment portfolios. To this end I evaluated whether EA generated portfolios as a profitable investment.

This investigation involves a portfolio search in the Mexican Stock Market, using data from January 4, 2016, to December 31, 2019. The data contains the daily closing prices of securities listed in this market. I was able to retrieve the data for 3431 different securities for that period. Using this data set to create different portfolios with at least 2 assets but a maximum of 100 assets, taking into account the weighted distributions, results in a continuous search space with infinite possibilities.

However, if we narrow down the search to only have integer fractions of 100 (to use 1% increments in the weights), the possible combinations become

$$C = \frac{(100 + 3431 - 1)!}{100! * (3431 - 1)!} \approx 7.3457 \times 10^{169}$$

Making it hard and computationally expensive to evaluate all portfolios individually.

Therefore, to create optimal portfolios, two optimization algorithms were applied:

1. Genetic Algorithm (GA)
2. Multi-Objective Evolutionary Algorithm based on Decomposition (MOEA/D)

Evolutionary Algorithms are a family of stochastic search techniques inspired by biological evolution. A group of candidate solutions to an optimization problem are recombined, mutated and evaluated, to iteratively generate better solutions. EAs have proven to be effective for large and complex search spaces[11][7][5], as is portfolio optimization[5]. The Genetic Algorithm (GA) and the Multi-Objective Algorithm based on Decomposition are examples of evolutionary algorithms.

GA has been around for over 50 years[3]. It was the first evolutionary algorithm developed and while it is outdated and falls short for solving Multi-Objective Problems (MOPs), it still serves as a benchmark, as it is fast and extensible[3].

MOEA/D is a framework that was first introduced in 2007[4], as an alternative to the Non-dominated Sorting Genetic Algorithm II (NSGAI) for solving multi-objective problems (MOPs)[4][7]. It is a simple but powerful algorithm that produces good results in a variety of problems[7], that combined with the flexibility and space in the framework for improvement, makes it a great research tool in the scientific community. Specifically for portfolio optimization, it has been proven to have good results[5][22].

This thesis is an investigation of using Evolutionary Algorithms for portfolio optimization. In particular we:

1. Generate non-dominated portfolios using MOEA/D and GA;
2. Evaluate the performance of these generated portfolios using simulations;
3. Compare the performance of the algorithms against well-known market benchmarks;
4. Explore the potential of using these algorithms for real-life investing.

The layout of this thesis is as follows: Chapter 2 will discuss the theoretical background of the portfolio optimization problem as well as the background of the methods used, followed by a literature review; Chapter 3 contains all the relevant information for implementing the solutions to the problem described; Chapter 4 contains a description for the experimental tests as well as the results; finally, in Chapter 5 conclusions are presented, summarizing the main findings of this work as well as its limitations, and including suggestions for future research.

## 2. Theoretical Framework

The final objective of this research is to automatically create efficient portfolios. To achieve that objective, we first needed to review all the theoretical background. As explained in the introduction, portfolio optimization is a problem relevant to two areas of knowledge: Computer Science and Finance. More specifically Portfolio Management and in this particular case, Artificial Intelligence (A.I.). This research will focus mostly on applying computer science techniques to optimize portfolios, using different financial models.

In this chapter, we will summarize the theoretical framework of this research. This review will explain the selection of methods and tools used to solve the problem. The first part (section 2.1) will focus on a formal definition of the problem using a model created by economists. The following sections will explore possible ways of creating and optimizing portfolios.

Finally, in this section, there will be a literature review exploring the most relevant state-of-the-art solutions to this problem.

### 2.1 Investment Theory

This section explores the developments that have occurred in the field of investment theory. The theories developed in this field provide a base for portfolio optimization. Markowitz was the first academic to visualize investment portfolios as an optimization problem. After that, improvements have been made to Markowitz's theory by other academics, such as William Sharpe. Portfolio theories roughly follow the same path, trying to model risks and return. In this section, Markowitz's theory as well as the Capital Asset Pricing Model will be explained in depth.

#### 2.1.1 Mean-Variance Portfolios

In 1952, Harry Markowitz provided a framework for evaluating portfolios in terms of the variance and the means of the return of the assets. This theory provided the first rigorous measure of risk for investors and showed how one selects alternative assets to diversify and reduce the risk of a portfolio[2].

He defines the return  $R_t$  of an individual asset at a given time  $t$  in terms of its price  $P$  as:

$$R_t = \frac{P_t}{P_{t-1}} - 1 \quad [1][2]$$

This can be seen as the percentage change of an asset's price from one time to another. This same equation can be applied to a portfolio using the total value of the portfolio over time.

The return of a portfolio is the weighted sum of the mean return of each asset. So if we have a portfolio of  $n$  assets, each asset will have a relative weight in the portfolio  $\omega_i$  and a mean return  $\mu_i$ , our expected return  $E[R]$  of the portfolio is given by:

$$E[R] = \mu = \sum_{i=0}^n \omega_i \mu_i$$

In this model, the risk of an asset is calculated as the variance of the returns and the risk of a portfolio can be calculated with a sum of covariances of each pair of assets. So, if the variance of the return of an asset is  $\sigma_i^2$  and the covariance between two given assets is  $\sigma_{i,j}$ , then the variance of the return of a portfolio can be expressed as:

$$\sigma^2 = \sum_{i=1}^n \sum_{j=1}^n \sigma_{i,j} \omega_i \omega_j$$

Given the two measurements explained above ( $\sigma^2$  and  $\mu$ ), Markowitz defines portfolio selection as an optimization problem, where  $A$  is the risk aversion of the investor:

$$\min(\sigma^2 - A\mu)$$

The risk aversion is defined by the investor, it says how much risk the investor is willing to take. A lower  $A$  value results in a portfolio with smaller variance and when  $A \rightarrow \infty$ , it means the investor only cares about returns.

Given a set of assets, if we plot each portfolios in *Risk vs. Return* ( $\sigma^2$  and  $\mu$ ) plane, this will result in a curve (Figure 1). The portfolios with maximum return given a set of assets and a risk are optimal portfolios. The **Efficient Frontier** is a line where the optimal portfolios are found.

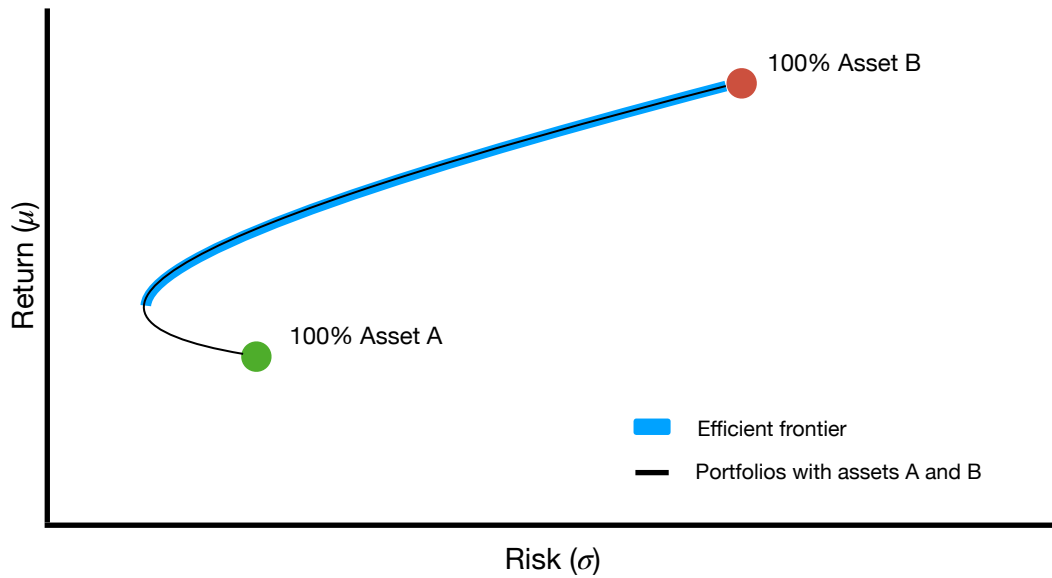


Figure 1. The Efficient Frontier of portfolios with two assets, A and B.

## 2.1.2 Capital Asset Pricing Model

The Mean-Variance model, though widely used in academia, falls short to represent reality. Mean-Variance offers an incomplete explanation for the relationship between the risk and the return. Capital Asset Pricing Model (CAPM), was developed shortly after the Mean-Variance model. The CAPM extends portfolio theory in a way that allows investors to evaluate the risk-return trade-off for both diversified portfolios and individual securities.

To do this, CAPM adds the concept of a Risk-Free asset to Markowitz's risky portfolios and provides a framework to analyze expected returns. A risk-free asset in practical terms is usually a government bond with a fixed return rate.

Another innovation in this model is introduction of systematic risk and unsystematic risk. The first one refers to the general market risk, conditions in the whole market that affect all individual assets. The unsystematic risk refers to the particular risk of an individual stock. The coefficient beta is a measure of the systematic risk of an asset compared to the whole market. It can be estimated with the following equation:

$$\beta = \frac{\text{cov}(R_i, R_m)}{\text{var}(R_m)}$$

$R_i$  = Return of an individual asset

$R_m$  = Return of the overall market

The beta coefficient represents the risk that cannot be mitigated with diversification.

Using this beta coefficient we can calculate the return that we should expect from an asset at a level of risk:

$$E[R_i] = R_{rf} + \beta_i(R_m - R_{rf})$$

where:

$R_{rf}$  = Risk-free rate

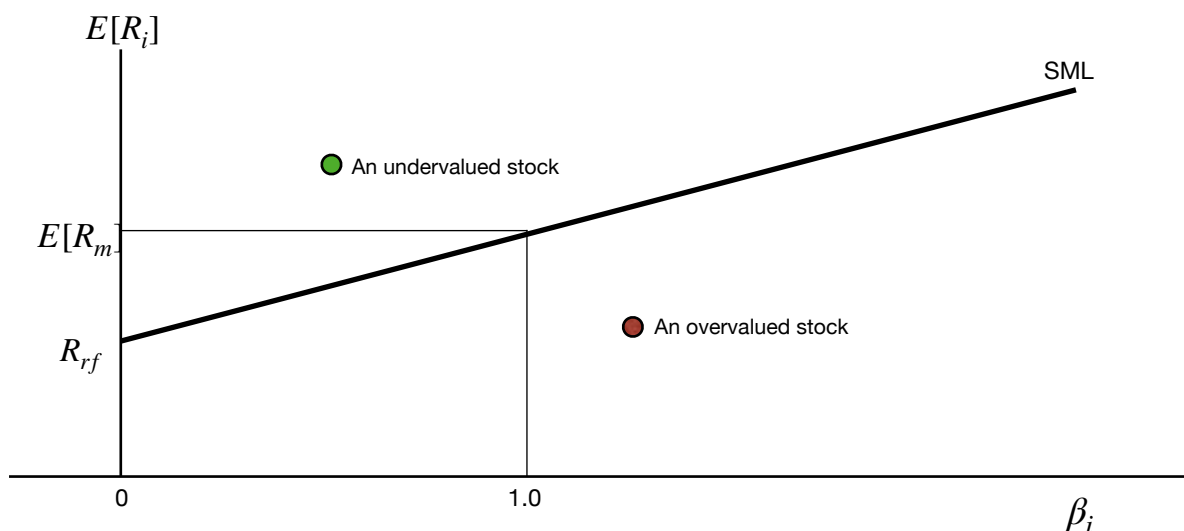


Figure 2. Chart showing the SML.

With these two measures (Beta and Expected Return), the CAPM model can be expressed as a chart using the security market line (SML). This line shows the average trade-off between risk and returns in the market. It is a straight line that intersects the vertical line (zero risk) at the free risk rate. As can be seen in Figure 2, securities with a higher return than the SML (for a given Beta), are undervalued according to this model. While securities below the SML, are overvalued.

In this table there are some guidelines to read different values of  $\beta$ :

Table 1: Guidelines for reading  $\beta$  values

$\beta < -1$	Asset's price moves in the opposite direction, and in a greater amount than the negative of the market
$-1 < \beta < 0$	Asset's price movement is in the opposite direction of the market
$\beta = 0$	Asset's price movement is uncorrelated to the market
$0 < \beta < 1$	Asset's price moves in the same direction, but in a lesser amount than the market
$\beta = 1$	Asset's price moves in the same direction and in the same amount as the market
$\beta > 1$	Asset's price moves in the same direction, but in a greater amount than the market

## 2.2 Traditional Methods for Portfolio Creation

The two models above allow us to evaluate and compare portfolios and individual securities. However we still need a way to create the portfolios. To do this we follow two steps[2]:

1. First we select assets;
2. then we optimize the portfolio.

With CAPM and Mean-Variance Model we can then evaluate the created portfolios.

### 2.2.1 Selecting Assets

Is the process of finding the assets that are best for the objective. This process involves conducting research and selecting assets based on mainly two characteristics:

1. Sector: The industry a company belongs to can help diversify assets, normal diversification uses assets from different industries;
2. Capitalization: Companies of different sizes have different behavior in the stock market. Bigger companies tend to be more stable while emerging companies can be very volatile.



With this information, an investor can do a technical analysis and determine in which securities the investment is best placed. Additionally, the objective of the investor is going to define how the assets are mixed and distributed. In particular, this objective will define the risk tolerance that a portfolio has balanced to an expected return.

## 2.2.2 Optimizing the Portfolio

Once a set of assets to invest in is decided, the next step is to decide the weight of each asset in the portfolio to either maximize returns and/or reduce risks. Throughout the years, there have been different methods and attempts to optimize a portfolio successfully, like Markowitz's Efficient Frontier explained before.

## 2.3 Evolutionary Algorithms

The Evolutionary Algorithms (EA) are a set of meta-heuristic algorithms, inspired by Darwin's natural selection[3], often used to solve optimization problems. In these algorithms, a set of candidate solutions to a problem are selected, combined, and altered to find new and potentially better solutions to the problem. Each candidate solution, has a value of *fitness*, which is a quantitative measure of how good is the solution at solving the problem.

Evolutionary Algorithms started with the appearance of the Genetic Algorithm, published by John Holland in the 1960's. Later in the 80's the Genetic Algorithm grew in popularity and many variants started appearing including multi-objective versions of the genetic algorithm as is the Non-dominated Sorting Genetic Algorithm (NSGA) and more recently the Multi-Objective Evolutionary Algorithm based on Decomposition (MOEA/D).

A solution in the context of evolutionary algorithms is often referred to as an *individual*. A group of individuals is called *population*. A population at a given point in the execution of the algorithm is referred to as a *generation*.

Solutions are represented as a binary string of a fixed size. Each position in the array is called *gene*, and each gene represents a decision variable. Newer implementations have used other types of strings as integers or floating point numbers.

The functions used to select, combine, and alter individuals are called *genetic operators*. These operators are:

1. Crossover: the *crossover* is a small function that takes two solutions and combines them to produce two other solutions that resemble the original solutions.
2. Mutation: similar to crossover, the *mutation* is a small function that takes a solution and makes a small alteration to it, producing a new solution.
3. Selection: selection in evolutionary algorithms is a stochastic process to draw solutions from a population based on their fitness.

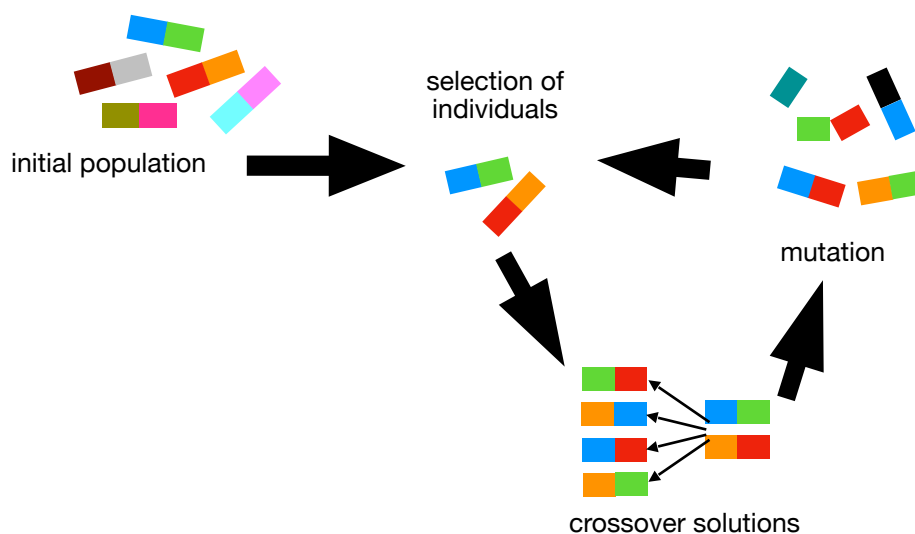


Figure 3. Basic flow of any evolutionary algorithm.

On a bigger scale, an evolutionary algorithm is a process that takes as input a problem and iteratively creates and improves a population of solutions to find an optimal[3]. At each iteration, solutions are drawn from the population using the *selection operator*. These solutions are then combined and altered using the *crossover operator* and the *mutation operator* to produce a new and improved population. This process is pictured in figure 3.

To successfully apply any given evolutionary algorithm to a problem some things need to be determined[3]:

1. A way to represent a solution as an array of a fixed size; this is also known as encoding of a solution.
2. A method to generate an initial population that is as diverse as possible.

3. Fitness function(s) to evaluate solutions.
4. **Genetic operators** to use: one crossover, one selection and one mutation operator.

In the following sections I will cover in more depth two flavors of evolutionary algorithms used in this research: the Genetic Algorithm (GA) and the Multi-Objective Evolutionary Algorithm based on Decomposition (MOEA/D).

### 2.3.1 Genetic Algorithm

The Genetic Algorithm was the first evolutionary algorithm, it was published in 1970[3]. It was created by John Holland[3]. Even though the algorithm is now outdated, it has provided the basis for every other evolutionary algorithm. One limitation of this algorithm is the lack of support for multi-objective problems.

Over the years the Genetic Algorithm has been used for portfolio optimization. It has proven to be effective in portfolio optimization using Markowitz Model [10]. More recently, S. Lim et al., in 2020 proposed an ensemble method based on GA, using more complex analysis, achieving good results using stock from S&P500 and KOSPI200 Indexes[9].

The algorithm works in the following way[3]:

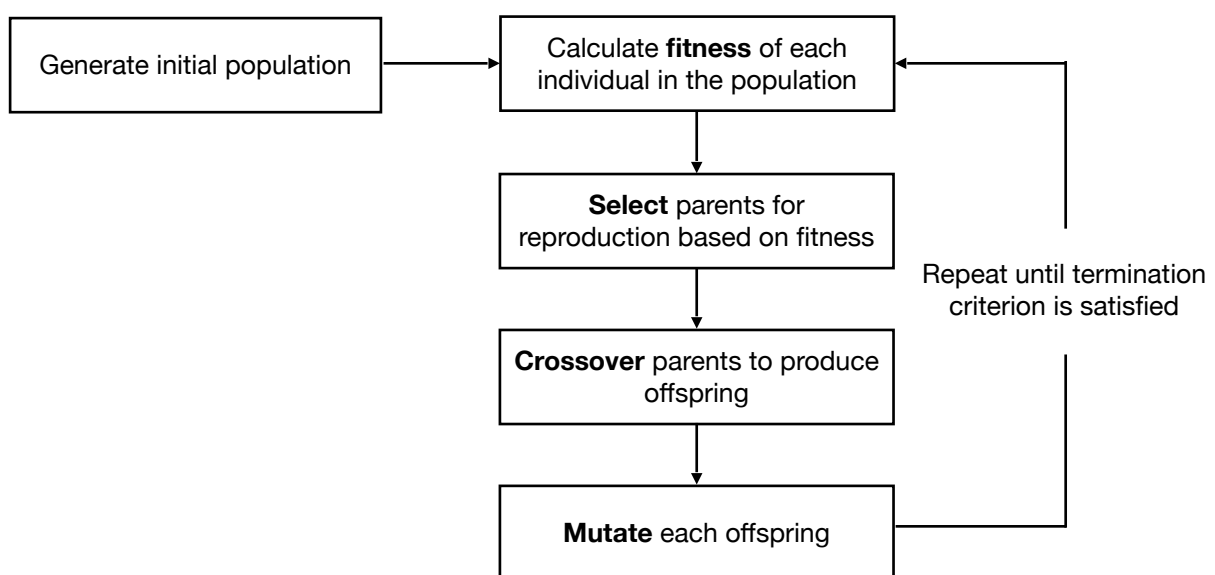


Figure 4. Genetic Algorithm

## **Fitness**

The fitness of an individual is a measurement of how good it is at solving our problem. Since we are talking about optimization we are trying to find not only a feasible solution, we are trying to find the best solution in the search space. A good fitness function will ensure that the GA finds the best solution available.

When we talk about portfolio optimization, the fitness function has to be a measurement of return and risk. In later sections, this will be discussed in more depth.

## **Generate Initial Population**

Generating an initial population is a challenge in the implementation of the genetic algorithm. The initialization method is mostly determined by the problem that we are trying to solve and the restrictions that come with it. It is crucial as it can also determine the areas of search space that the algorithm will cover. A common practice for this step is to simply generate random arrays that meet all requirements imposed by the problem.

To generate the initial population I created a method specific for portfolio optimization. To produce a solution, the initialization method will first select few random stock, then it will look for the stock that correlate less with the randomly selected stock to achieve some diversification.

## **Selection**

This is the process for selecting which solutions to crossover. It usually involves selecting parents with a probability based on fitness however many different approaches can be used such as tournament selection, roulette wheel selection, or rank selection. The purpose of the selection is to eventually eliminate low-quality individuals while still keeping diversity.

## **Crossover**

The crossover is a function that takes two individuals and combines them into two new individuals that resemble the originals. The idea is that if two different individuals are good solutions to a problem if we combine them, we may find an even better solution. Also for the crossover, there are many approaches and different algorithms for combining individuals, the crossover that is used is dependent on the constraints of the encoding, but also different crossovers have a different effect on each generation. Crossover is used to explore new areas in the search space.

## **Mutation**

The mutation is applied to a single individual, it makes small alterations on it to generate a new individual. Mutations are used to create more diversity in the population, it is a complement of the crossover operator, while crossover makes big leaps in the search space, mutation serves for fine-tuning and also adding new elements to the current genes in the whole population.

### **2.3.2 Multi-Objective EA Based on Decomposition**

Multi-Objective Evolutionary Algorithm based on Decomposition, abbreviated as MOEA/D, was developed by Qingfu Zhang and Hui Li and published in 2007[4]. The idea is to use mathematical decomposition to address multi-objective problems with an evolutionary algorithm[4]. A multi-dimensional problem is decomposed into many problems of a single dimension, using uniformly distributed vectors. Since it was published there have been several extensions made to enhance this algorithm, making it a relevant up-to-date framework[5][7].

On a bigger scale, MOEA/D follows the same flow as the genetic algorithm, the main difference is the support for multi-objective problems. In general terms, MOEA/D generates many different fitness functions using decomposition, and the algorithm tries to optimize the population to each of the fitness functions. To achieve this, the population is divided into neighborhoods based on areas of the solution space. With the decomposition and the use of neighborhoods, MOEA/D can find high-quality solutions distributed across the solution space.

MOEA/D uses a vector decomposition to handle multi-dimensional problems. A multi-dimensional problem is decomposed into many uni-dimensional problems by using unitary vectors that are uniformly distributed across the solution space. In a two-dimensional solution space (bi-objective problem):

$$\vec{\lambda} = (x, y)^T \quad |\vec{\lambda}| = 1$$

In this figure a bi-dimensional problem is decomposed into 5 different problems, each of them identified with a lambda vector. Each of those vectors can be seen as an independent unidimensional objective to be optimized.

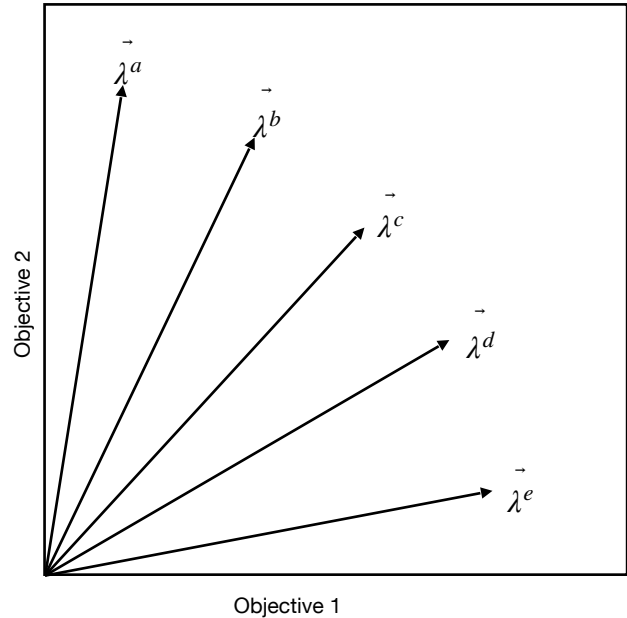


Figure 5. Decomposition

To convert a multi-dimensional fitness to any of the decomposed objectives, we select a lambda vector and a decomposition method such as a weighted sum:

$$\text{maximize } g(x | \lambda) = \sum_{j=1}^m \lambda_j f_j(x)$$

where:

$m$  is the number of objectives

$\lambda_j$  is the  $j$ th component of the vector  $\lambda$

$f_j(x)$  is the  $j$ th fitness function (objective)

Another decomposition method, presented as an alternative to the weighted sum, is the Tchebycheff approach, the principle is similar, to calculate fitness using the  $\lambda$  vectors:

$$\text{minimize } g(x | \lambda, z^*) = \max_{1 \leq j \leq m} \{ \lambda_j | f_j(x) - z_j^* | \}$$

where:

$z^*$  is a reference point.

A simple method for setting the reference point  $z^*$ , is to use the best value in the population for each objective:

$$z^* = (z_1^*, \dots, z_j^*)^T$$

$$z_i^* = \max\{f_i(x) \mid x \in \Omega\}$$

where:

$\Omega$  is the current population.

In both approaches, when calculating the fitness of a solution, there is a fitness value that corresponds to each of the vectors. Each function (created with each vector) is a unidimensional problem to be solved. The vectors are distributed uniformly to guide the search in different directions across the solution space, resulting in a varied set of solutions evenly distributed.

If we have a solution  $x^c$ , and  $x^c$  is the fittest solution so far using the weighted fitness with vector  $\lambda^c$ .

If we set neighborhood size to 3, then the best solutions for  $\lambda^b$  and  $\lambda^d$  are neighbors of  $x^c$ .

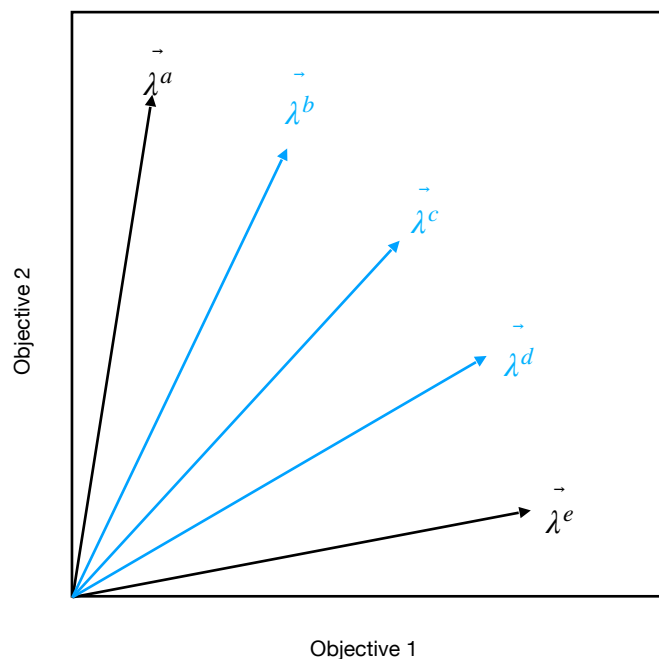


Figure 6. Neighborhoods

To keep diversity in each generation and to be able to improve each of the generated fitness functions, the MOEA/D applies selection depending on the area in the solution space. To do this, the concept of neighborhoods is applied. A neighborhood is a group of solutions close to each other (when plotted in the solution space). To calculate neighborhoods, all solutions are sorted and MOEA/D keeps only the best individual of each decomposed problem, then, to know which solutions are nearby, we simply use the euclidean distance between the lambda vectors (figure 6).

To use MOEA/D we need to define/create the following:

1. Multi-Objective Problem:  $f^1(x), f^2(x), \dots$
2. Population size:  $N$
3. A set of weight vectors, containing  $N$  vectors:  $P = \{\lambda^1, \lambda^2, \dots, \lambda^N\}$
4. Neighborhood size:  $T$
5. Stopping criteria.

During each generation, MOEA/D will maintain a set of ordered solutions (population)  $\Omega = \{x^1, x^2, \dots, x^N\}$  where  $x^i$  is the fittest solution found so far for the weight vector  $\lambda^i$ , and a set of non-dominated solutions  $EP \subset \Omega$ .



---

**Algorithm 1:** MOEA/D [4]

**Output:**  $EP$ , a set of non-dominated solutions.

**1. Initialization:**

Create initial population  $\Omega = \{x^1, \dots, x^N\}$ .

Calculate the distance between weight vectors to find  $T$  closest vectors to each vector. Create sets  $B(i) = \{i_1, \dots, i_T\}$ , where  $\lambda^{i_1}, \dots, \lambda^{i_T}$  are the  $T$  closest vectors to  $\lambda^i$ .

Set  $EP = \emptyset$ .

Initialize reference point  $z^*$ .

**2. Update population:**

**for**  $i = 1$  **until**  $i = N$

Select two indexes  $k, l$  from  $B(i)$ , and generate solution  $y$  with crossover from  $x^k, x^l$ .

Apply mutation to  $y$  with a probability.

Calculate the fitness of  $y$  using each weight vector  $\lambda^j \in P$ . If  $f(y | \lambda^j) \geq f(x^j | \lambda^j)$ , then replace  $x^j$  with  $y$  in  $\Omega$ .

Add  $y$  to  $EP$  if no solution from  $EP$  dominates  $y$ . Remove all solutions from  $EP$  that are dominated by  $y$ .

Update reference point  $z^*$ .

**3. End** if stopping criteria is satisfied. Otherwise, go to **step 2**.

---

Shortly after MOEA/D was published, some improvements were made[5][7][13][14]. One successful attempt was adapting MOEA/D to use Differential Evolution (DE) operators[7]. Differential Evolution is explained in more depth in the next section.

### 2.3.3 Differential Evolution

The Differential Evolution (DE) is yet another heuristic that optimizes a problem by improving a population of candidate solutions. To improve candidate solutions, DE uses simple formulas to combine existing solutions and to generate new solutions.

The crucial idea behind DE is a scheme for generating new solutions[27]. DE generates new solutions by adding a weighted difference vector between two candidate solutions to a third one.

Given 3 different candidate solutions  $\hat{x}_1$ ,  $\hat{x}_2$  and  $\hat{x}_3$ , a new solution  $\hat{x}'$  is generated according to

$$\hat{x}' = \hat{x}_1 + F \cdot (\hat{x}_2 - \hat{x}_3)$$

where  $F$  is a real and constant factor that controls the amplification of the differential vector  $(\hat{x}_2 - \hat{x}_3)$ , subject to  $F > 0$ .

Differential Evolution was first published in 1997, as a single objective optimization algorithm and long before MOEA/D first appeared. In the time of its inception the authors, R. Storn and K. Price, demonstrated that DE was capable of minimizing continuous space functions, and proved that it was superior to Adaptive Simulated Annealing (ASA) as well as Annealed Nelder & Mead approach[27].

Later, in 2007, Chung Kwan, Fan Yang, and Che Chang proposed the replacement of mutation and crossover operators of the NSGA-II with a variant of differential evolution (DE). Proving that for the real world problems, NSGAI-DE generated better results than NSGA-II[28]. Consequently, in 2009, Hui Li and Qingfu Zhang created a version of MOEA/D that uses a variant of differential evolution. The operators proposed for MOEA/D Differential Evolution are the following[7]:

Given three candidate solutions on the same neighborhood  $\hat{x}_1$ ,  $\hat{x}_2$  and  $\hat{x}_3$ , a new solution  $\hat{y}$  is generated according to:

$$\hat{y} = \begin{cases} \hat{x}_1 + F \cdot (\hat{x}_2 - \hat{x}_3) & \text{with probability } CR \\ \hat{x}_1 & \text{with probability } 1 - CR \end{cases}$$

Then the polynomial mutation generates  $\hat{y}' = (y'_1, y'_2, \dots, y'_n)$  from  $\hat{y}$  in the following way:

$$y'_k = \begin{cases} y_k + \sigma_k \cdot (b_k - a_k) & \text{with probability } P_m \\ y_k & \text{with probability } 1 - P_m \end{cases}$$

Experiments using these methods proved MOEA/D to be superior to the Elitist Non-Dominated Sorting Genetic Algorithm (NSGA-II)[7]. This is due to MOEA/D's ability to do parallel searches in different parts of the plane, and with the support of DE operators, the algorithm is effective even in complicated Pareto fronts[7].

## **2.4 Literature Review**

Ever since Harry Markowitz set the basis for portfolio theory as an optimization problem, there have been attempts to achieve optimal asset distribution. In the last years, using computational intelligence for portfolio optimization has been a field of interest for both finance and computer science. To get familiar with the current practices and to get to know the background framing for this investigation, I conducted a literature review. In this review, I selected the most relevant state-of-art solutions to portfolio optimization. Each of these solutions was analyzed and classified by publication date, optimization algorithm, portfolio model, and evaluation strategies.

Given the scope of this review, I divided the content into two sections: single-objective and multi-objective optimization. Later followed by some conclusions drawn from this review.

### **2.4.1 Single Objective Portfolio Optimization**

In 2006, W. Chen, R. Zhang, Y. Cai, and F. Xu, proved that Particle Swarm Optimization (PSO) is an effective algorithm for portfolio optimization. Their approach worked with a constrained Mean-Variance model with transaction costs[15]. This result was later improved by Jianguo Cao and Liang Tao (2010), who applied a mutation to the solutions in the algorithm to enhance results[18].

In that same year, 2010, Hoklie and L. R. Zuhail published a paper about portfolio optimization using a Genetic Algorithm. After conducting some experiments they conclude that a Genetic Algorithm can produce good portfolios if it is correctly configured[10]. Some months later in 2010, A. Talebi, M. A. Molaei, and M. J. Sheikh concluded that a Genetic Algorithm is more effective for portfolio optimization than PSO by running simulations with portfolios generated by each method[19].

In 2020, S. Lim, M. Kim, and C. W. Ahn created an ensemble method based on a Genetic Algorithm to find optimal portfolios. This method has the innovation of using different models: Mean-Variance, CAPM, and Momentum Strategy; optimized by a Genetic Algorithm. The ensemble method was evaluated by the Return of Investment in the long term (5 years), having positive results[9].

#### **2.4.2 Multi-Objective Portfolio Optimization**

For multi-objective portfolio optimization, algorithms seem more varied. There is a constant discussion on which algorithm performs best.

M. Chen, Jian Weng, and Xia Li (2009) attempted to find optimal portfolios using many algorithms Multi-Objective Extremal Optimization (MOEO), Non-dominated Sorting Genetic Algorithm II (NSGA-II), Strength Pareto Evolutionary Algorithm 2 (SPEA2) and Pareto Archived Evolution Strategy (PAES). They compared the results based on the Coverage and Front Spread, demonstrating that NSGA-II finds more varied solutions and from higher quality[16].

Another study from that same year comparing Multi-Objective Particle Swarm Optimization (MOPSO), NSGA-II, SPEA2, and Parallel Single Front Genetic Algorithm (PSFGA) for portfolio optimization was published by S. K. Mishra, G. Panda, and S. Meher. In this study, MOPSO produced higher quality portfolios, more evenly spread in the Pareto front than the rest of the algorithms[17].

Two studies from Divya Kumar, K.K.Mishra, (2017) and R. Ramadhiani, M. Yan, G. F. Hertono, and B. D. Handari (2018), tested Multi-Objective Co-Variance based Artificial Bee Colony (M-CABC) for portfolio optimization[21][23]. The conclusions from that study are that M-CABC produces good results and is capable of accommodating constraints as needed[23] but other algorithms can produce better results[21].

In a recent study, MOEA/D was presented as the best performing algorithm for portfolio optimization (B. Y. Qu, Q. Zhou, J. M. Xiao, J. J. Liang, and P. N. Suganthan, 2017)[22]. In this paper MOEA/D was compared with Multi-Objective Differential Evolution (MODE), NSGA-II, Non-Dominated Sorting Algorithm (NDS) and Multi-Objective Comprehensive Learning Particle Swarm Optimizer (MOCLPSO), in an extensive experiment with Mean-Variance portfolio optimization. The results showed that MOEA/D with objective normalization finds higher quality portfolios than the rest of the algorithms in both dimensions (risk and return)[22]. A different study from 2018 (H. Zhang, Y. Zhao, F. Wang, A. Zhang, P. Yang, and X. Shen) also applied MOEA/D to portfolio optimization successfully, the authors proposed an initialization method for the weight vectors with better results than the standard version[5].

Outside of portfolio optimization, MOEA/D has been the subject of study in other papers achieving good results. Hui Li and Qingfu Zhang, the original creators of MOEA/D[4] did an extensive study after the initial paper was released[7]. In this paper the authors test MOEA/D against NSGA-II, demonstrating in a variety of problems, that MOEA/D can be superior to NSGA-II[7]. In this paper, the authors also propose a new variation of MOEA/D, based on differential evolution (DE)[7], evidence suggests that decomposition methods are very promising for multi-objective optimization in evolutionary algorithms. Another paper from Ishibuchi, H., Doi, K. & Nojima, Y. (2017) explained the importance of objective normalization in multi-objective normalization, and by applying normalization, the authors can obtain better results from MOEA/D, making it a more robust algorithm[13].

### **2.4.3 Conclusions of this Review**

In this review, there are references to more than eight different optimization algorithms. Most of them regarding multi-objective optimization. There is a constant discussion trying to find a better algorithm for the purpose and the options are more varied. Using multiple objectives for portfolio optimization opens up more possibilities to model the problem and comparing results is a difficult task by itself. In the near past NSGA-II appeared to be the norm in multi-objective optimization, but in recent years, works on MOEA/D have proven it to be a good alternative. As it grows in popularity, the community creates new improvements to the algorithm, making it more suitable for portfolio optimization.

For single-objective portfolio optimization, the situation is a bit different,

the choices are either PSO or GA. When it comes to performance, both of them seem to be pretty close. The open question here remains to be: How can we evaluate portfolios with a single function?

If we talk about portfolio models used, all researchers tend to use the Mean-Variance model, probably because of its simplicity. When the studies try to replicate real-life decisions more variables can be accommodated to the Mean-Variance model, such as transaction costs[15], cardinality constraints[15][21], or metrics coming from another model like CAPM[9] or Risk Budgeting Strategy[20].

#### **2.4.4 Algorithms Found in Literature** (more details in Appendix 1)

- Particle Swarm Optimization[15][18]
- Genetic Algorithm[10][9][19]
- Multi-Objective Extremal Optimization[16]
- Multi-Objective Particle Swarm Optimization[22][17]
- Multi-Objective Differential Evolution[22][16]
- Multi-Objective Co-variance based Artificial Bee Colony[23][21]
- Non-dominated Sorting Genetic Algorithm II[22][17][16]
- Multi-Objective Evolutionary Algorithm Based on Decomposition[22][7][5][13]

### 3. Methodology

For a portfolio to be successful, the stock included must have an expectancy to grow over time, so that later, when the stock is sold, a profit is made. To achieve such, each stock must be analyzed carefully, however it is nearly impossible to individually assess each market factor. In the earlier section, we have explored different analytical theories for evaluating and measuring assets and risks, as well as some strategies to mitigate some of these risks. Additionally, we reviewed methods in computer science to find optimal combinations automatically.

#### 3.1 Tools and Methods

By using the analytical models and the evolutionary algorithms as optimization frameworks, the computer can create investment portfolios based on historical performance. Having this in mind I created a method for generating and evaluating portfolios. The idea is to combine different approaches from our theoretical framework to find the optimal combination of algorithms and financial models.

By selecting one optimization algorithm and one financial model at a time, this framework will generate a portfolio or a set of portfolios, based on historical data.

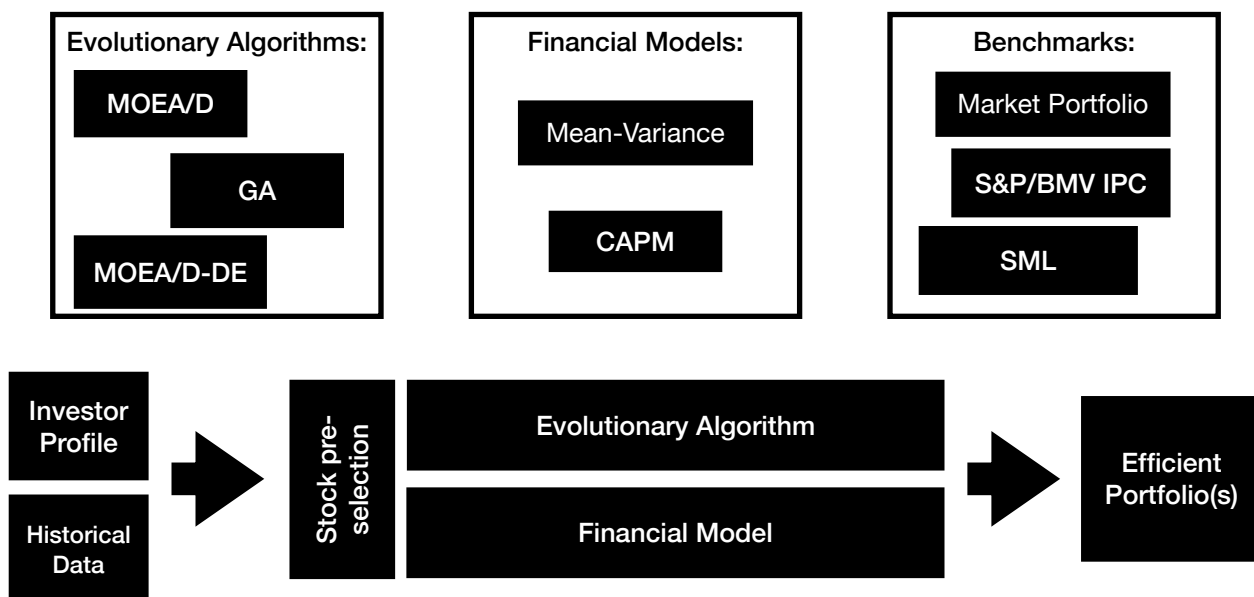


Figure 7. Interaction of the components in the solution. In this framework, only one evolutionary algorithm is selected at the time as well as only one financial model to produce a set of portfolios.

## 3.2 Implementation

This section will explain thoroughly my implementation of each of the components seen in part 3.1.

As seen in the theoretical framework, to be able to generate portfolios we need to define these aspects: encoding, initialization, fitness function(s), and genetic operators. In this section, I will cover each of the components that will make these evolutionary algorithms excel in portfolio selection.

Out of a full set of stocks, these EAs will return a subset of stocks where the investor should invest in. Additionally, the EAs will output the weights for each of the stocks. First I will cover single-objective optimization, using a simple Genetic Algorithm, followed by Multi-Objective optimization, using MOEA/D.

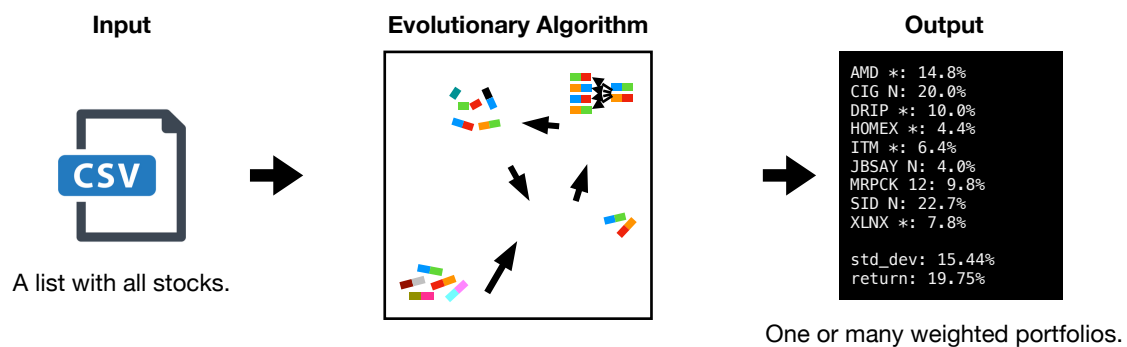


Figure 8. Input and output of the Evolutionary Algorithms.

Lastly, to improve the results of the evolutionary algorithms, I created an algorithm to select *the best* assets. The algorithm is explained in more depth in section 3.2.3 (Algorithm 3). This algorithm will serve as a filter before using the Evolutionary Algorithms for optimization. Using the mean return, standard deviation and the coefficient of variation the algorithm will select a subset of stocks to then be fed to the Evolutionary Algorithms. This will narrow the search space, improving execution times by removing noise from the security list, such as inactive companies listed in the stock market.



## 3.2.1 Genetic Algorithm

### Fitness Function

The fitness function will guide our algorithm to find better portfolios, a fitness function that represents the reality better will result in portfolios that perform better in real life. In section 2.1 we explored different portfolio models to evaluate assets. For this investigation, the fitness functions will be derived from that theoretical frame, in particular from Capital Asset Price Model[26] and the Mean-Variance Model[1]. The focus is to optimize for low-risk and high-return portfolios using a single fitness function (objective).

In this scenario, I have seen different approaches, usually involving a combination of metrics. The two approaches that fit my research best are:

$$1) \quad f = \frac{R_p}{Risk}$$

$$2) \quad f = \frac{(R_p - R_f)}{Risk} + CAPM$$

In the equations above  $R_p$  refers to the expected return (or mean return) of the portfolio and  $R_f$  is the risk-free rate. The first one is a simple ratio between risk and return. The second equation contains two different concepts, the first term is the Sharpe Ratio, which tells us is an investment is good when comparing the risk taken to the return of the risk-free asset, the second term is the valuation using the Capital Asset Pricing Model, which will let us know if we are buying an undervalued or overvalued asset.

### Encoding

Defining the encoding to be used is the first step in development. For genetic algorithms, it is necessary to use an encoding that is easy to interpret, but also simple to evaluate with fitness functions. Also, it is important that the encoding is as free as possible, having a very restrictive encoding can lead to producing inadmissible solutions. Producing inadmissible solutions can have effects on computational performance but also the quality of the results.

The number of possible assets available for investing is finite and constant. The obvious encoding is to have an array with one space for the weight of each asset. So if we have  $N$  possible assets, a solution can be represented as:

$$[\omega_1, \omega_2, \dots, \omega_N]$$

where  $\omega_i$  is the weight of the asset identified with index  $i$  in the portfolio. Such that

$$\sum_{k=1}^N \omega_k = 1$$

Assets not included in the portfolio will simply have  $\omega_j = 0$ .

However, this encoding has some drawbacks, to generate a random solution we need to make sure the sum of weights is always 1. This also applies when we use mutation and crossover operators.

By removing the constrain of the sum of weights, we have an encoding with relative weights that is more abstract, but easier to handle. To find the weight of an asset expressed as a percentage, we simply apply a normalization.

Given a solution  $S$ :

$$S = [c_1, c_2, \dots, c_N]$$

$$\omega_i = \frac{c_i}{\sum_{k=1}^N c_k}$$

This approach produces an overhead when computing the fitness of the solutions, but it reduces the complexity of generating, crossing, and mutating solutions.

To avoid going to extremes or to impossible situations I created two additional constraints to the model:

- Use arrays of integers, where each element is between 0 and 100. This gives enough granularity for the weights but one unit difference is still noticeable in the portfolio.

- The minimum number of assets in a portfolio is 2 and the maximum is 100. The number of assets in a portfolio is determined by the costs of holding and the amount of money invested. However, in this study transaction/holding costs are not being considered. I have set the maximum number to 100 because studies show that in a market of 2000 securities, 99% of the diversifiable risk is eliminated with portfolios of 95 assets (Gordon Y. N. Tang, 2003).[12]

## Initialization

Using random initialization can have a large impact on the last generations of an evolutionary algorithm. To avoid inconsistent results and to start with a better population of portfolios in the first generation, I created an initialization method appropriate for portfolio design.

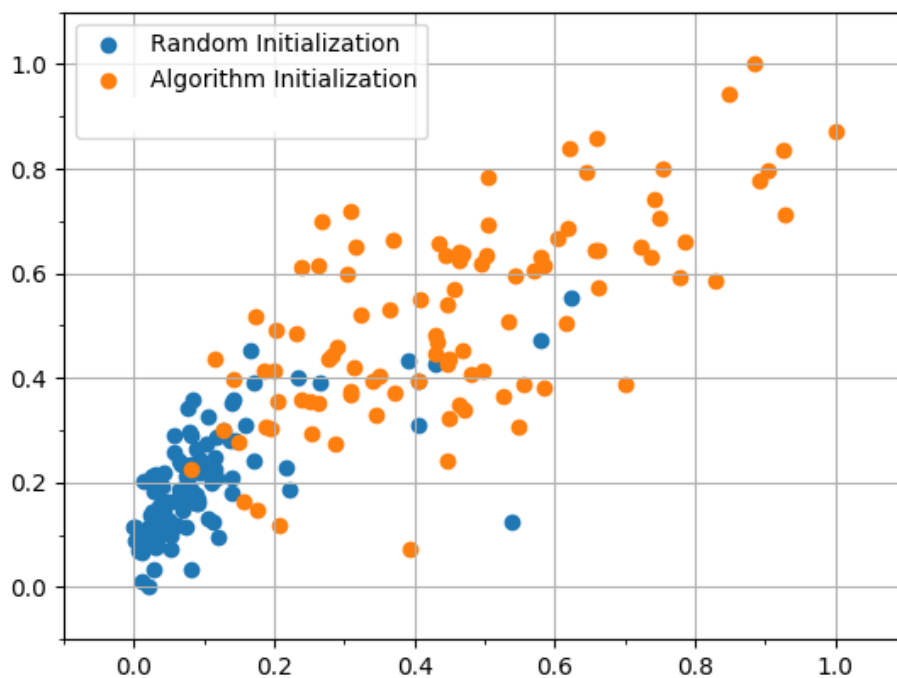


Figure 9. Comparison between a random initialization and the proposed initialization (Algorithm 2). The Y-axis corresponds to return and the X-axis corresponds to the risk.

The initialization algorithm will produce a set of solutions. To produce each solution, the algorithm will first select random stock. And the rest of the stock is selected based on the least correlation. The weights of each stock in the portfolio are random and the quantity of different stocks in each portfolio is also random, so for it to work properly a minimum and a maximum number of stocks must be defined (marked as  $a$  and  $b$  in the algorithm). Empirical tests showed better results than generating a population randomly (figure 9).

---

**Algorithm 2:** Create an initial set of portfolios

---

**Input:** Population size ( $N$ ).

**Output:** A random set of solutions ( $\Omega$ ).

**for**  $i = 1$  until  $i = N$

Generate a random number  $K$ , where  $a \leq K \leq b$ .

Pick randomly  $K \div 4$  different stocks and place them into a new portfolio  $P$ .

**while** the size of  $P < K$

**for** each stock  $S$  in  $P$

Find stock least correlated to  $S$  that is not in  $P$ .

Add selected stock to  $P$ .

Assign a random weight to each stock  $S \in P$ .

Add  $P$  to  $\Omega$ .

---

## Genetic Operators

For the Genetic Algorithm, we need to select three genetic operators: Selection, Crossover, and Mutation.

To select the operators I followed the implementation made by S. Lim et al[9]. For selection, a tournament is used to reduce early convergence without re-scaling. Moreover, tournament selection is expected to have a better takeover time, compared to the proportional selection methods (such as roulette and rank selections).

Table 2: Genetic Operators for GA

Algorithm	Selection	Mutation	Crossover
Genetic Algorithm	Tournament Selection	Single-Point Mutation	Two-Point Crossover

### 3.2.2 MOEA/D

#### Fitness Function

In multi-objective optimization, there is no need to create a single metric that

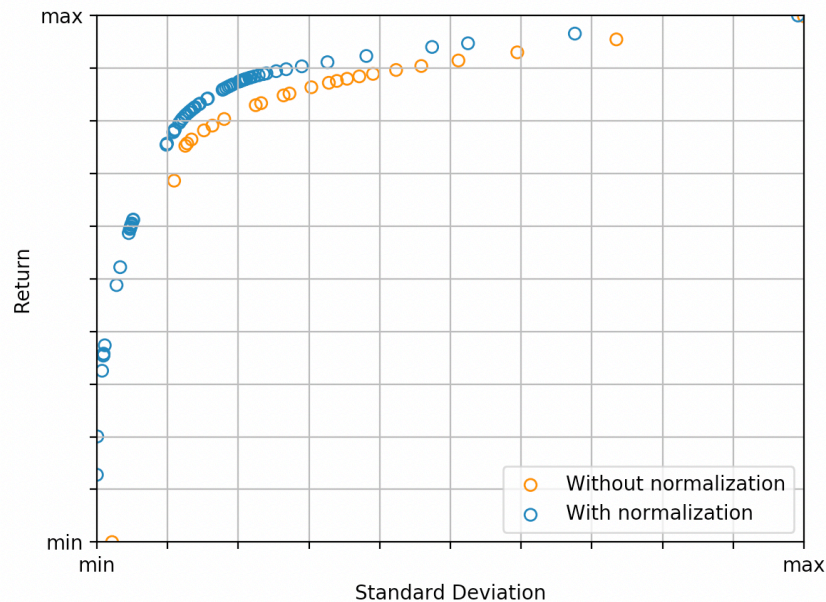


Figure 10. Solutions found by two executions of MOEA/D with the same configuration except for objective normalization. In this plot we can clearly appreciate a better distribution of the solutions in the Pareto front when using normalization.

correlates with the risk and the return. However, in this research, I reviewed two different ways to calculate the risk:

1. Using the Mean-Variance model, the risk is calculated using the portfolio standard deviation formula.
2. Using CAPM, we calculate the Beta of the portfolio.

The second objective in this optimization is the return. Again we can use two different calculations depending on the model we use:

1. Mean-Variance Model: The return is calculated using the historical mean return of the portfolio.
2. CAPM Valuation: Calculate the return using the historical mean return and using the Expected return equation. The difference between those values tells us if the investment is better or worse than the market average (overvalued or undervalued asset).

### Genetic Operators

Selection with MOEA/D is given by the algorithm itself, so it is only necessary to select a mutation and a crossover. When we talk about MOEA/D-DE, then all operators are given, no need to select any other operator.

Table 3. MOEA/D and MOEA/D Genetic Operators

Algorithm	Selection	Mutation	Crossover
MOEA/D	MOEA/D Selection	Single-Point Mutation	Two-Point Crossover
MOEA/D-DE	MOEA/D-DE Selection	Polynomial Mutation	Differential Evolution

### Objective Normalization

When dealing with multiple objectives, one of the problems that arise is that each objective has its scale, the results in MOEA/D prioritizing one of the objectives more than the rest. For example, if returns of portfolios range between 10% and 15% and the standard deviation for those same portfolios is in the range between 8% and 13%, MOEA/D will favor return more than standard deviation, this will create difficulties when we need to distribute solutions evenly across the plane. This behavior is explained thoroughly in the paper by Ishibuchi, H., Doi, K. & Nojima, Y. (2017)[13].

To mitigate this behavior, normalization is applied to the objectives[13], this will result in having all objectives on the same relative scale. For this project, I will be using the same normalization as in the paper (Ishibuchi et al. 2017). The fitness corresponding to  $i$ th objective  $f(x)_i$ , will be normalized as:

$$z_i = \frac{z_i - z_{iL}}{z_{iU} - z_{iL} + \epsilon}$$

calculated using an upper and lower reference point ( $Z_{iL}$  and  $Z_{iU}$  respectively). It is a min-max normalization function with a little modification: to avoid the risk of the denominator becoming zero, a positive constant  $\epsilon$  is added,  $\epsilon = 10^{-6}$ .

This method proved to have positive results with the data of this research (figure 10).

### 3.2.3 Stock Pre-Selection

During the creation of this system, the first problem to be encountered is how to deal with a big amount of stock options. Having a large search space can affect widely the performance of an evolutionary algorithm, it increments the generations needed to converge as well as the size of the solutions and the population. A pre-selection of the stock was implemented to make the algorithm leaner and faster. It takes into consideration some aspects:

1. Mean return
2. Standard deviation
3. Variance covariance matrix

Using those characteristics, this program will select a subset of stocks to then be fed to the Evolutionary Algorithms. The number of stocks selected to be used will be set to be around 500. This implies that the solutions of the Evolutionary Algorithms are arrays of 500 integers.

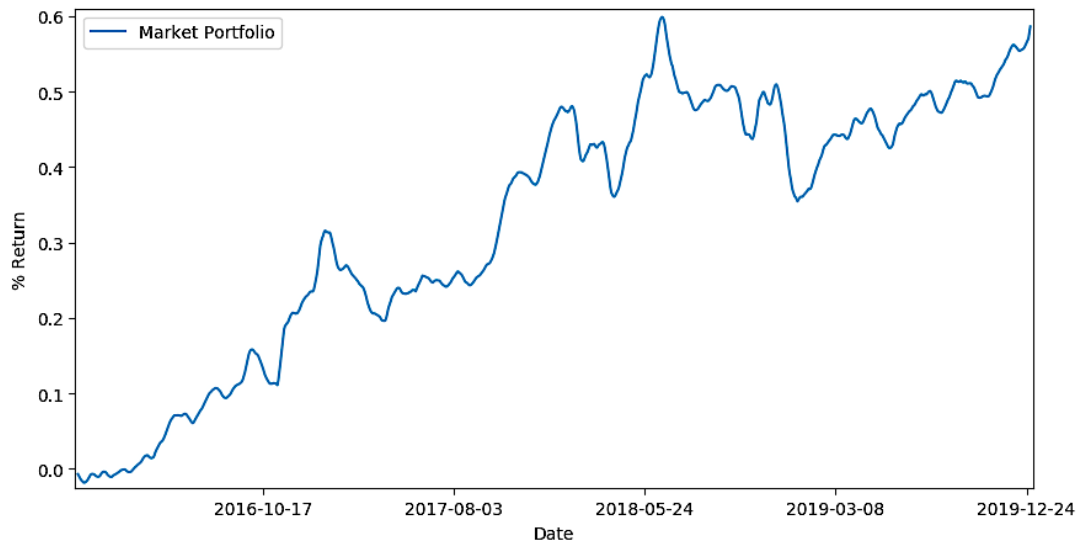


Figure 11. Mexican Stock Exchange Market Portfolio percentage returns.

---

**Algorithm 3:** Stock Pre-selection

---

**Input:** Set of stocks ( $S$ )

**Output:** Subset of stocks ( $S'$ )

Remove all the stocks whose standard deviation ( $\sigma$ ) is zero.

Remove all outliers regarding the return:

Calculate the mean return of the population ( $\mu_p$ ).

Calculate the standard deviation of the mean return of the population ( $\sigma_{\mu p}$ ).

Remove all the stocks with a return  $\mu$  outside of the interval:  $[0, \mu_p + 3\sigma_{\mu p}]$

Return all remaining stocks as  $S'$

---



## 4. Experimental Phase

The objective of this investigation is to find a suitable framework for Portfolio Selection and to compare it against current existing methods used in finance. The idea is to use Evolutionary Algorithms for creating portfolios using historical prices. Once we have created the portfolios we evaluate them using future prices of the assets. In this section, I will explain how the performance of the selected methods will be evaluated.

### 4.1 Data

Originally all data was provided by Bolsa Institucional de Valores (BIVA). BIVA is a relatively new stock exchange in Mexico, it started operations in July 2018, and they kindly supported this project by providing all daily closing prices for their stock from the beginning of their operations, on July 25, 2018, until the 31st of December of 2019. In addition to the closing prices, they also provided the capitalization of each stock. This price list is composed of daily closing prices for 3431 different stocks that were traded at that time.

Additionally, given the need for more data, I extended the dataset to 4 years worth of data, starting from 2016 until the end of 2019. The additional data was gathered from BMV (Bolsa Mexicana de Valores) and Yahoo Finance. However, not all of the stocks were listed since the beginning and/or until the end of the period. So, to compare stock equally in the whole period, I removed all the stocks that were added to the list after January 4th, 2016, and all the stocks that were removed from the stock exchange before December 31, 2019, as well as all the stock with missing data. After filtering the data, the list is narrowed down to 1503 stock with exactly 1005 close prices. Those 1005 prices correspond to each of the business days in the time range in observation.

Using the capitalization and the daily price changes, we can produce a market portfolio. The market portfolio is a portfolio that includes all assets weighted by their capitalization (figure 11).

For the experiment, I divided the data into two sets of equal size, by date, one dataset for the model to select portfolios, and the other dataset to evaluate portfolios.

The first dataset, to create the portfolios contains 754 closing prices, which correspond to the first 3 years (2016, 2017, and 2018). The dataset for evaluating the portfolios contains the last year of data or exactly 251 closing prices. Both sets contain the same securities. As explained in the introduction, the entire dataset goes from January 4, 2016, until the 31st of December of 2019, and it is constituted by 1503 different securities that were available throughout the whole time range.



Figure 12. Data split.

## 4.2 Metrics

To decide whether a portfolio is better than others, I selected some metrics:

### CAPM Expected Return:

$$E[R_p] = R_{rf} + \beta_p(R_m - R_{rf})$$

### Historical Mean Return:

$$\mu = R_p = \sum_{i=0}^n \omega_i \mu_i$$

### Beta:

$$\beta = \frac{cov(R_p, R_m)}{var(R_m)}$$

### Standard Deviation:

$$\sigma = \sqrt{\sum_{i=1}^n \sum_{j=1}^n \sigma_{i,j} \omega_i \omega_j}$$

The first two metrics are different calculations for the expected return of investment[2] and the last two reflect measurements of risk systematic and unsystematic[2]. These metrics can provide expectations for the future performance of a portfolio.

I will gather the metrics for the *creation dataset* and the *evaluation dataset* separately.

## 4.3 Benchmarks

Once all the portfolios have been generated using the methods explained in the previous chapter and having them evaluated using the metrics selected, I compared them side by side. To have an idea of how these portfolios are performing I selected three benchmarks: the security market line, the market portfolio, and the Mexican stock market (BMV because of its initials in Spanish *Bolsa Mexicana de Valores*) S&P index.

These benchmarks and the portfolios will be evaluated using the evaluation dataset, so we can compare future performance.

### Market Portfolio

The idea is that all investors are highly intelligent, so creating a portfolio containing all the stocks in the market, weighted by capitalization can give us an idea of how much money are the investors making on this market. However, it is impossible to invest in all stocks in the market, so this is a theoretical reference. Having that in mind, this portfolio can give us an expectation of how should our portfolios behave.

### BMV S&P Index

This index was created by financial experts, selecting stocks that are representative of this particular market. This index gives us an idea of how the market is moving, it serves as a proxy for the Market Portfolio, however, this index is a realistic portfolio, as it only contains a fraction of the companies in the market.

## SML

This line shows the average trade-off between risk and return of investment. Any portfolio above this line can be treated as a good portfolio, portfolios on this line are average performing portfolios and the ones below the line are worse than average portfolios. We can obtain this chart by plotting the CAPM equation:

$$E[R_i] = R_{rf} + \beta_i(R_m - R_{rf})$$

## 4.4 Procedure

To gather the results I followed these steps:

**Step 1:** Find optimal parameters for each of the algorithms (explained below in section 4.4.1):

- A. Mutation Rates (GA, MOEA/D & MOEA/D-DE)
- B. Crossover Rates (GA, MOEA/D & MOEA/D-DE)
- C. Neighborhood Size (MOEA/D & MOEA/D-DE)
- D. Scaling Factor (MOEA/D-DE only)
- E. Distribution Index (MOEA/D-DE only)
- F. Initialization Method (GA, MOEA/D & MOEA/D-DE)
- G. Normalization (MOEA/D & MOEA/D-DE)
- H. Decomposition Method (MOEA/D & MOEA/D-DE)

**Step 2:** Create using *creation dataset* with the following methods:

- A. MOEA/D + Mean-Variance
- B. MOEA/D + CAPM
- C. MOEA/D-DE + Mean-Variance
- D. MOEA/D-DE + CAPM
- E. Genetic Algorithm + Mean-Variance
- F. Genetic Algorithm + CAPM

**Step 3:** Select the most significant portfolios at different risk levels (low and high). This only applies to MOEA/D and MOEA/D-DE. Solutions obtained from each of the algorithms will be ordered by ascending risk. Then a solution will be selected at the 15th percentile for low risk, and at the 80th percentile for a high-risk portfolio.

**Step 4:** Gather metrics (Mean Return, Standard Deviation, Expected Return, and Beta).

**Step 5:** Compare gathered metrics against the metrics calculated using future prices using *simulation dataset*.

**Step 6:** Compare results against benchmarks.

Please find the full algorithm configuration in Appendix 2.

### 4.4.1 Parameter Optimization

To find the optimal parameters I ran individual tests for each parameter and value.

**For each parameter:**

1. Select 5 different values for the selected parameter.
2. For each of those 5 values:
  - A. Run the algorithm 10 times with the selected value.
  - B. Collect metrics (Fitness, Standard Deviation, Number of Solutions Found)
3. Compare results and select value.

Please find full results for the optimal parameter search in Appendix 2 and 3.

## 4.5 Results and Discussion

In this section, divided into four parts, I will present the results obtained from the experiments and the subsequent discussion. In the first part, I will explain the performance of the algorithms and techniques used as optimization methods for portfolios creation. In the second part, I will explore the risk handling and result predicting capacity of the models. In the third part, I will compare the results against well-known benchmarks. Finally, in the fourth part, I will present a brief discussion of the results.

## 4.5.1 Optimization Comparison

The purpose of this section is to analyze which algorithm serves better for portfolio optimization. Each algorithm was used twice, first with the Mean-Variance model and then with the CAPM model. It is important to note that the ability of each algorithm to find optimal solutions is related to the shape of the Pareto front, and these Pareto fronts are themselves determined by the fitness functions we use. Changes in the portfolio model, result in different behaviors of a given algorithm.

### 4.5.1.1 Mean-Variance Model

As seen in the literature, when using multi-objective optimization for the Mean-Variance portfolio, the Pareto front produces a curve that goes from Low-Risk-Low-Return to High-Risk-High-Return. In figure 13, the results for the three algorithms are in the same plot.

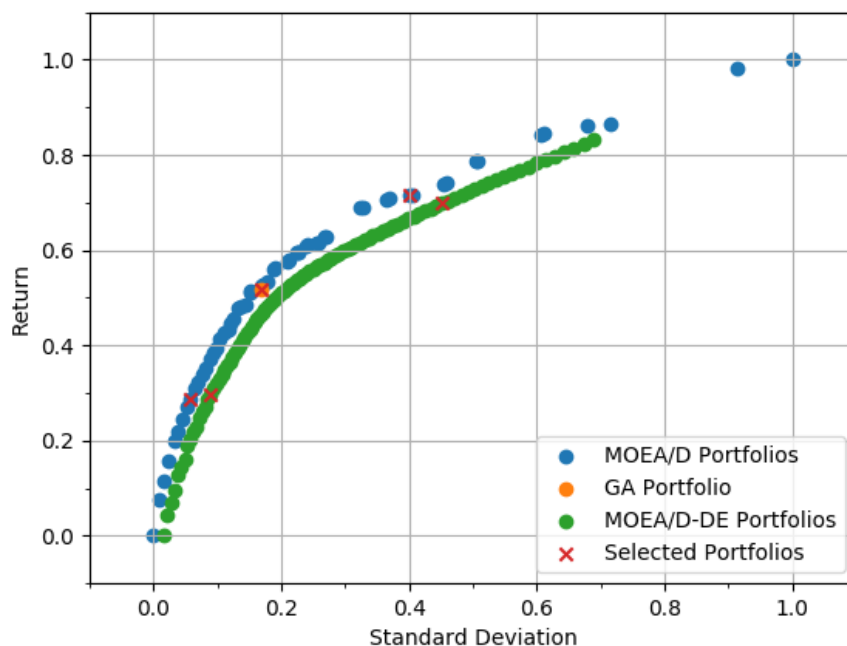


Figure 13. Solutions found by each algorithm when using a Mean-Variance Model. Please note that the Return and the Standard Deviation are normalized to values between 0 and 1 using a MinMax function. Selected Portfolios (marked with the red cross) refer to the portfolios used for later analysis and simulations.

Unquestionably, MOEA/D found the best results, stretching each portfolio for a higher return for any given level of risk. However, when the set of portfolios found by MOEA/D is compared directly to the set of portfolios found by MOEA/D-DE, we can see advantages in MOEA/D-DE even if the quality is a bit lower. The set of results of MOEA/D-DE are evenly spread across the lower and upper limits, conformed by 99 different points (portfolios). MOEA/D found fewer results (74 different portfolios), leaving bigger gaps between each portfolio. However, in the middle of the line, the portfolios are closer together.

Single Point Crossover and Swap Mutation operators tend to find higher quality results in fewer generations than Differential Evolution operators. However, in the longer term, the limits of Single Point Crossover and Swap Mutation are lower (figure 14).

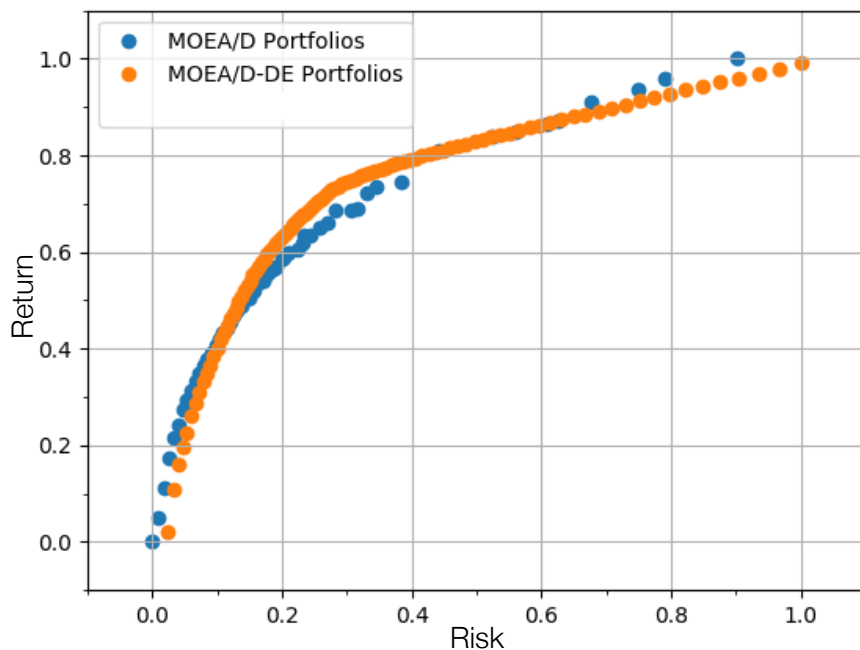


Figure 14. Solutions found by MOEA/D and MOEA/D-DE, when run for 1000 generations, 500 more generations than in the executions used for gathering results.

The Genetic Algorithm is a great option if we know what we are looking for, its implementation is faster and simpler than the multi-objective EAs, and the portfolio found has a similar quality than the ones found with the multi-objective algorithms. In this case, the fitness function pointed to the middle of the two objectives, giving balanced results. Further customization can be achieved when scaling the components in the fitness function, for a better risk and return balance.

In the three algorithms, there is room enough to accommodate more constraints for more robust searches.

#### 4.5.1.2 CAPM

The results of the CAPM optimization are not as straight-forward as the Mean-Variance's. In this setting, the MOEA/D-DE out-bested the other two algorithms. We can note that the negative behavior for the MOEA/D operators seen in the previous section is exaggerated when using CAPM equations. Results for MOEA/D are fewer and further apart. Also, it is important to note, that the set of solutions found by MOEA/D and MOEA/D-DE differ in shape.

The highest risk portfolio for MOEA/D is probably a lucky strike achieved by the genetic operators, finding a far away portfolio with better quality than the rest.

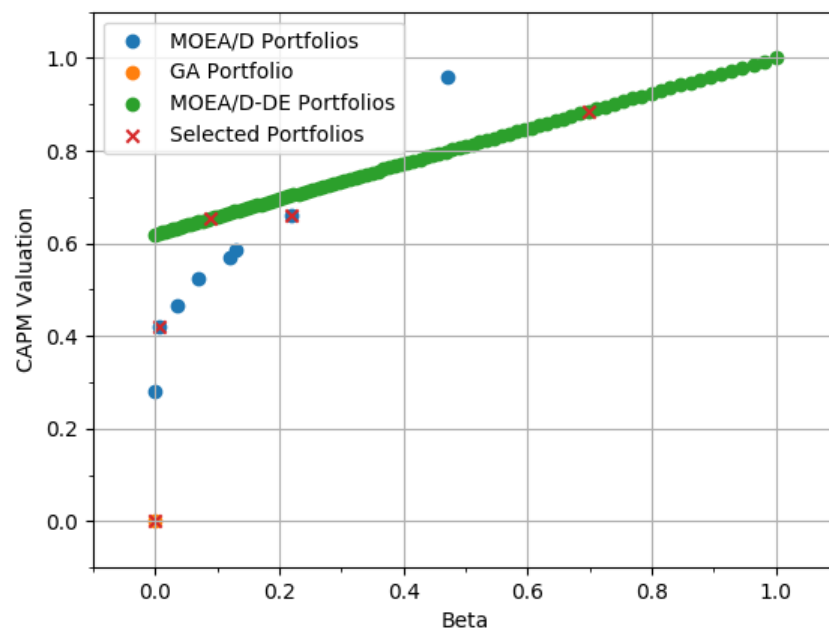


Figure 15. Solutions found by each algorithm when using a CAPM Model. Please note that the portfolio Beta and CAPM values are normalized between 0 and 1 using a MinMax function.

The Genetic Algorithm did not perform as expected. The objective of GA was to find a balanced portfolio, but it found an extremely low-risk-low-return portfolio (when compared to the ones found by MOEA/D and MOEA/D-DE).



## 4.5.2 Simulations and Portfolio Model Comparison

In this section, we will compare the performance of each portfolio found. This comparison will be based on the evaluation dataset. As explained before, to find optimal portfolios, I used historical data from 2016, 2017, and 2018. The portfolios found by the algorithms are close-to-optimal portfolios during that period, achieving the highest returns with a low standard deviation. However, it is uncertain if the trend can continue in the same direction, and if so, which characteristics we need to look at. To evaluate the models, we will look ahead into the future prices (daily 2019 prices), and compare the expectations with the actual behavior of the portfolios.

The results can be found in Appendix 4, where metrics by portfolio were gathered into a single table. Additionally, the assets contained and their weights for each portfolio can be found in Appendix 5.

### Genetic Algorithm

Two tests were executed using the Genetic Algorithm, one using the Mean-Variance Model, the second using the CAPM model. The Mean-Variance portfolio registered a historical mean return of 97% per year with a standard deviation of 38% and a Beta of 0.66. The CAPM portfolio registered a historical mean return of 33% per year with a standard deviation of 21% and a very low Beta of 0.000099. These values can be seen graphically in figure 16.

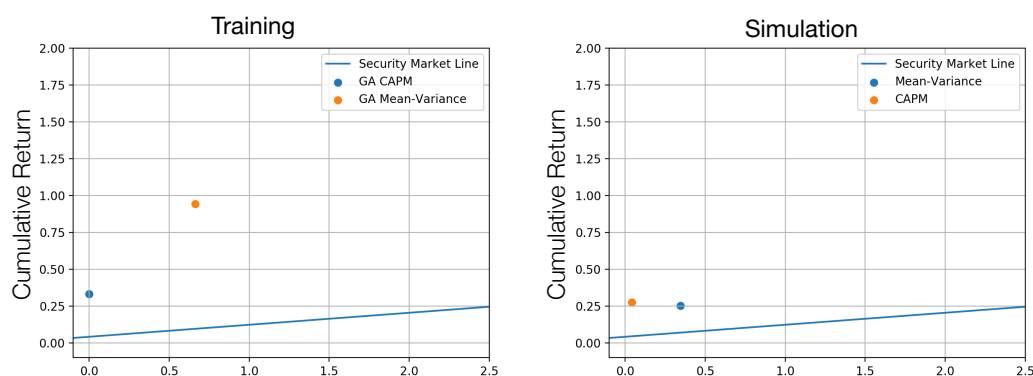


Figure 16. In the horizontal axis is portfolio Beta, and on the vertical axis is the return. In these two plots the GA portfolios are compared against the SML. On left-hand side, Betas and returns are calculated using data from 2016 to 2018 and on the right side, Betas and returns are from 2019.

When running the simulation, after a year, both portfolios achieved a very similar level of return (27% for CAPM portfolio and 25% for Mean-Variance portfolio). However, as it can be seen in figure no. 17, the Mean-Variance portfolio is more volatile. In that year, the Mean-Variance portfolio had a standard deviation of 19%, while the CAPM portfolio stayed as a more stable option with a standard deviation of 14%. Also, there was a bigger gap in the volatility explained by the beta coefficient. In 2019, we can observe a Beta of .04 in the CAPM portfolio, while the Mean-Variance portfolio had a Beta of .34 in that same period.

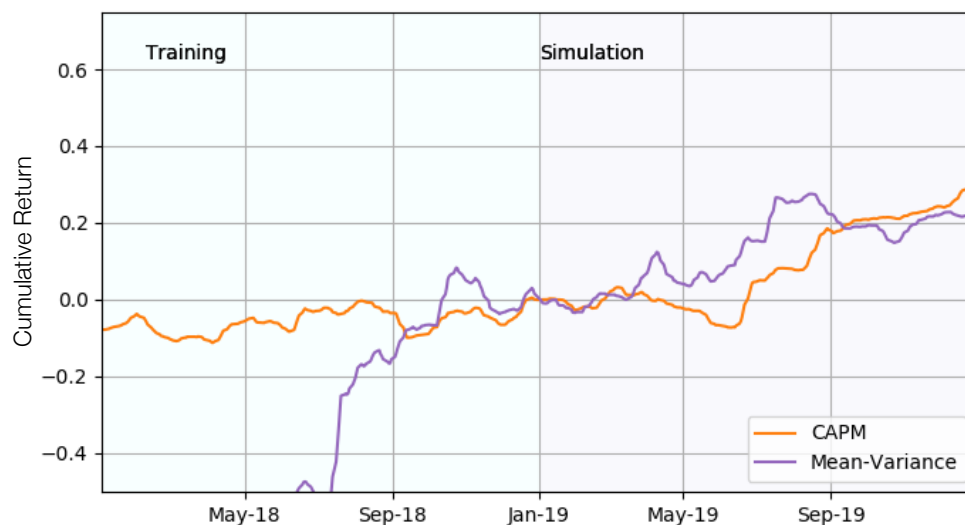


Figure 17. In this figure, the cumulative returns are plotted over time. Both lines cross the zero in 01-Jan-19, as if that day is the moment of investment.

## MOEA/D

The procedure for MOEA/D was slightly different. Same two tests were executed, but since MOEA/D can find a set of optimal solutions, I selected two portfolios from each execution at different percentiles (described in section 4.4) one for low risk and one for high risk. This results in four different portfolios, two CAPM portfolios, and two Mean-Variance portfolios (figure 18).

When we compare directly the CAPM Low-risk portfolio with the Mean-Variance Low-risk portfolio we see that CAPM found a portfolio with a very high return (92%) and a very high standard deviation (104%), but with an extremely low Beta of 0.004. However, the Mean-Variance portfolio has different characteristics: 59% mean annual return with a standard deviation of 16% and a Beta of 0.39. When we look at the values from the simulation, in 2019, the CAPM low-risk portfolio grew 31% and had a standard deviation of 19%, while the Mean-Variance portfolio only grew 10%.

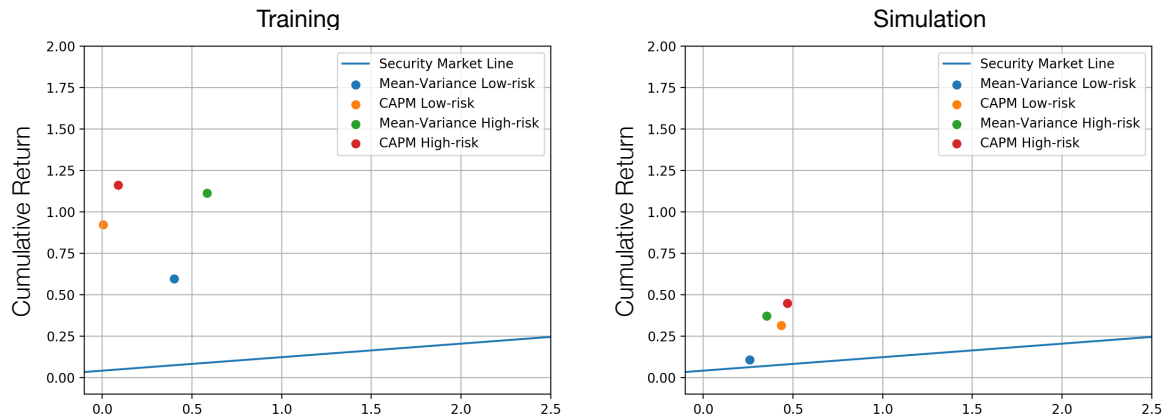


Figure 18. Returns (y-axis) against Beta (x-axis) for the MOEA/D portfolios. In blue, the Security Market Line is plotted. Left side corresponds to the creation dataset, right side to the evaluation dataset.

For the high-risk portfolios, the situation is not so different. CAPM found a portfolio with low-beta and high return, with a significantly higher standard deviation than the Mean-Variance portfolio (140% vs. 50% standard deviation) but a lower Beta (0.09 and 0.58 respectively). Then, during 2019, the CAPM portfolio out-bested the Mean-Variance portfolio when it comes to gross return.

In figure 19, we can see the Mean-Variance high-risk portfolio having a higher volatility than the rest of the portfolios, while the CAPM high-risk portfolio appears to have a more steady growth during the simulation period. The CAPM low-risk portfolio has a very similar line when compared to the CAPM high risk, but the variations are in a smaller amount, resulting in a lower growth rate.

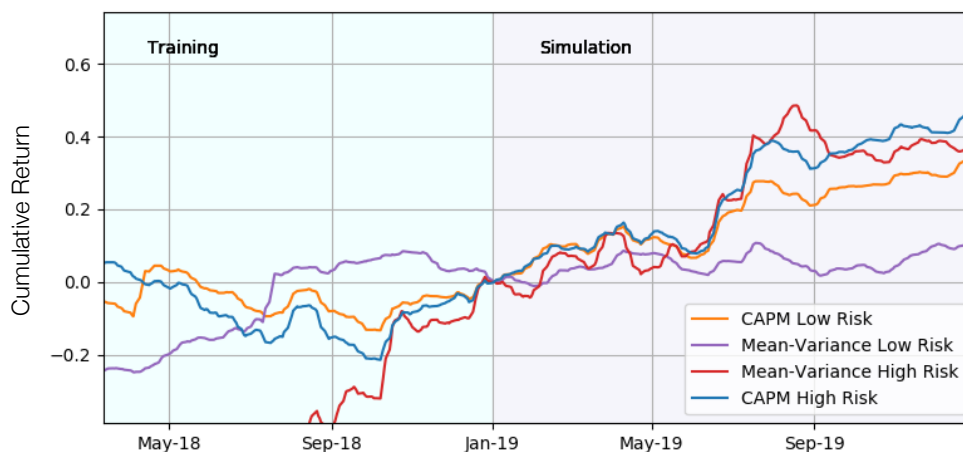


Figure 19. Cumulative returns are plotted over time for the MOEA/D portfolios.

## MOEA/D-DE

The result found by MOEA/D-DE does not differ too much from MOEA/D results. The CAPM portfolios have higher mean return rates (125% and 153%) at the cost of high standard deviation (193% and 294%), while the Mean-Variance portfolio have a more moderate return rate (61% and 114%) with moderate standard deviation (23% and 78%) but at higher Beta values when compared to the CAPM Betas which are close to zero. It is interesting to note that the Mean-Variance high-risk portfolio has a lower Beta (0.41) than the Mean-Variance low-risk portfolio which has a Beta of 0.88. These values can be seen plotted in figure 20.

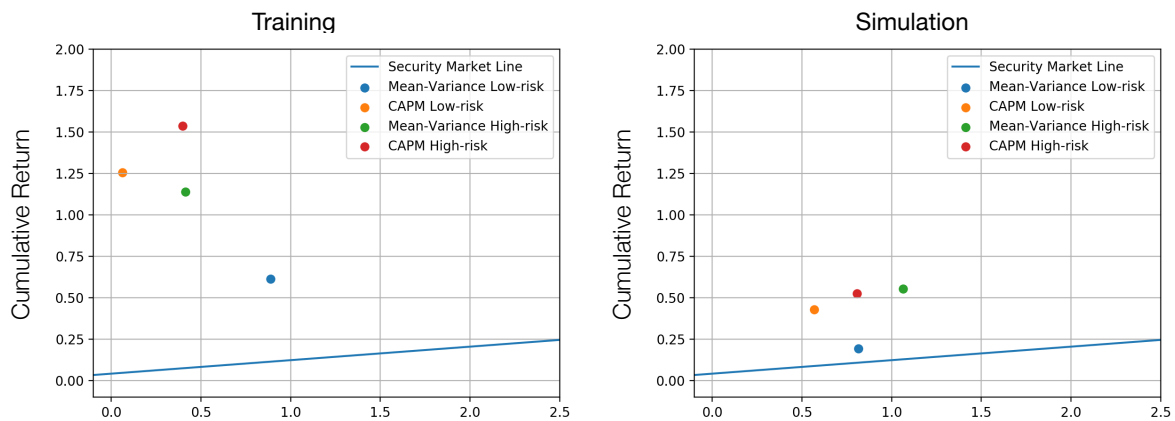


Figure 20. MOEA/D-DE portfolios compared against the SML. On left-hand side, Betas and returns are calculated using the creation dataset, on the right side using the evaluation dataset.

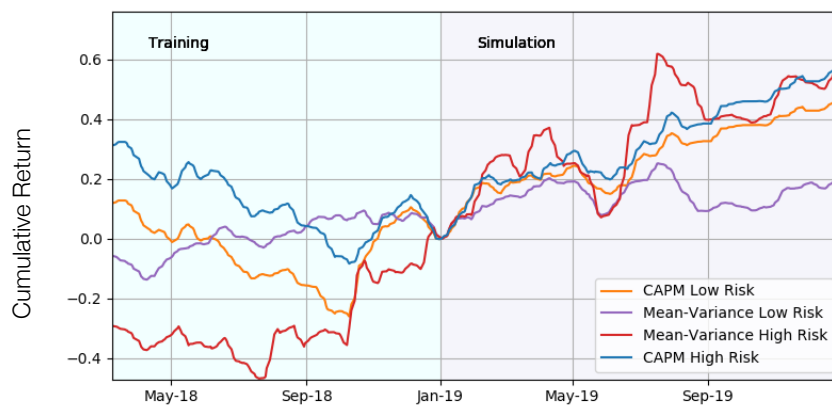


Figure 21. Cumulative returns are plotted over time for the MOEA/D-DE portfolios.

In figure 21, we can see a simulation of the cumulative return rates of each portfolio. Hard to see in the plot but the portfolio that produced the most return during that year was the Mean-Variance High-risk portfolio with 55% return, but closely followed by the CAPM High risk, with 52% gross return. As seen in the MOEA/D portfolios and contrary to my initial expectations the Mean-Variance high-risk portfolio had the highest volatility, 106%. Both CAPM portfolios have a similar shape, but the low-risk portfolio shows the variations in a lesser amount, reflected in the return.

### 4.5.3 Comparison Against Benchmarks

Finally, to assess the results I compared each of the portfolios against the selected benchmarks: Security Market Line, Market Portfolio, and BMV S&P Index. This comparison should tell us if the methods used are better or worse than the average investment in the market.

#### SML

As seen in the second chapter, plotting a portfolio against the SML tells us if it performs better or worse than the expected market average. A portfolio above the SML is an undervalued portfolio, which is giving more return than expected, and a portfolio under the SML is an overvalued portfolio. In figure 22, we can see all the portfolios obtained plotted against the SML with the values from the simulation period, from the 2nd of January, 2019 until the 31st of December, 2019.

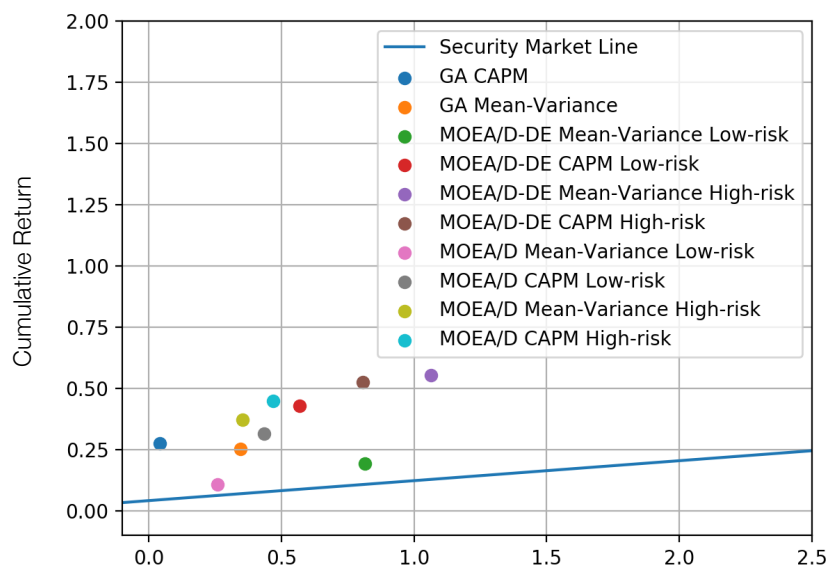


Figure 22. Returns during the simulation period (01/01/19 - 31/12/2019) on the y-axis and Beta coefficient in the x-axis. SML represented as a blue line.

As we can see, all of the portfolios generated more return than their expected return according to the CAPM model. The GA CAPM stands out, generating a return of 27% under a very low beta of 0.04.

### Market Portfolio

In figure 23, we can find a time series showing the cumulative returns of each portfolio and having the market portfolio as a comparison. All of the portfolios created with Evolutionary Algorithms produced more gross returns than the market portfolio except for MOEA/D MV Low Risk (brown line). However, the market portfolio appears as a line with steady growth and low volatility, while most of the portfolios are reasonably more volatile, in particular the MOEA/D-DE MV High-Risk portfolio (pink line). As a safer option, showing less volatility the MOEA/D-DE CAPM low-risk portfolio (light blue line) is one of the portfolios with the highest return.

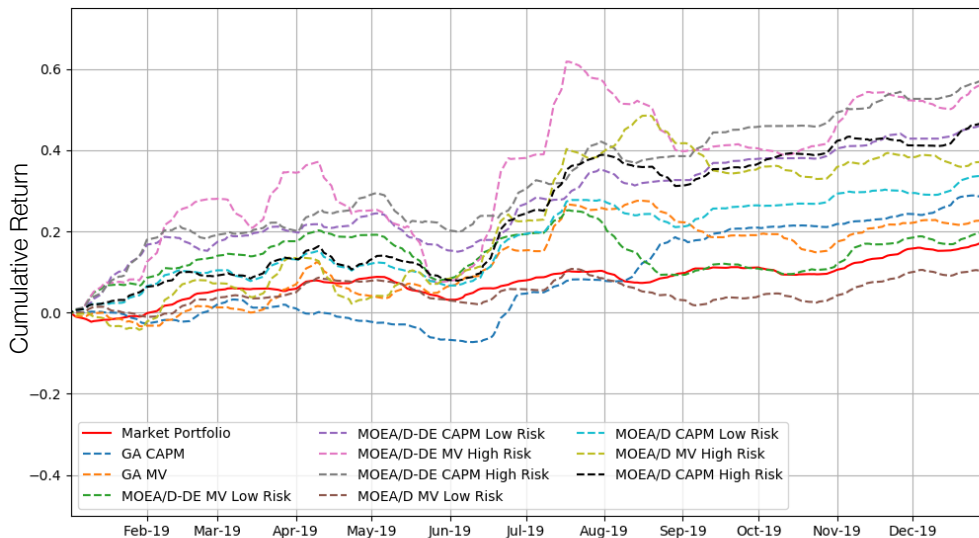


Figure 23. Cumulative returns of each portfolio (dashed lines) plotted next to the market portfolio (solid red line) over the time.

## BMV S&P Index

As seen previously, the market portfolio and SML are both theoretical concepts. In contrast, this benchmark is an attainable portfolio. It contains the most representative assets of the Mexican Stock Market (BMV) and it is used as a proxy of the Market Portfolio, one important difference between the two is that the BMV S&P Index is less volatile than the market portfolio. In figure 24 we can see the comparison of all portfolios (dashed lines) against the index (solid red line). All of the portfolios perform better than the BMV S&P Index, reporting higher returns at the end of the simulation period, on the 31st of December, 2019.

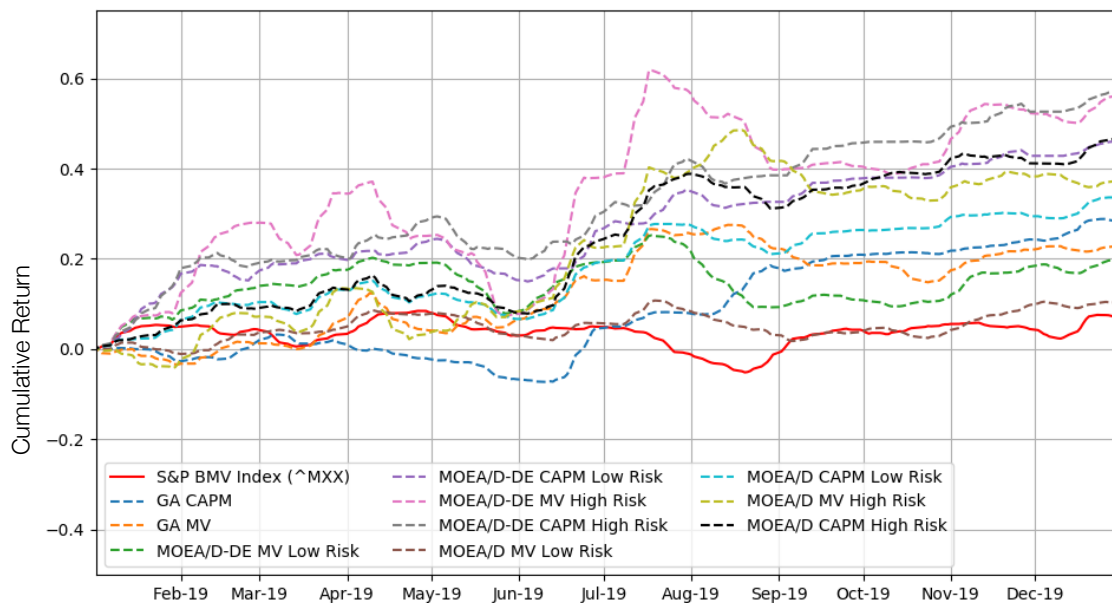


Figure 24. Cumulative returns of each portfolio (dashed lines) plotted next to the S&P BMV Index  $\wedge$ MXX (solid red line) over the time.

### 4.5.4 Discussion

After carefully reviewing the results, I found some important points:

1. CAPM together with the Evolutionary Algorithms appears to give the most consistent results, during the simulation year, the volatility remained steady and the growth was constant in the CAPM portfolios.
2. MOEA/D-DE is the most versatile optimization method used in the research, it delivers a wider range of results than its predecessor (MOEA/D) and more evenly spread.
3. While MOEA/D appears to find better results for Mean-Variance portfolios, evidence suggests MOEA/D-DE has the potential to surpass it.

4. When searching for optimal portfolios for the CAPM model, diversification is lost, the algorithms concentrated most of the investment into few assets, which may seem like a risky practice.
5. Contrary to the previous point, when optimizing for Mean-Variance portfolios, when trying to find a low-risk portfolio, the algorithms try to excessively diversify to lower the portfolio standard deviation, which in real life can have a large impact on the transaction costs.
6. Having in mind the two previous points, the Beta coefficient and standard deviation are complimentary risk measures, reflecting different behaviors of an asset.



## 5. Conclusion and Future Work

This research has been an investigation into the use of the Genetic Algorithm, MOEA/D, and MOEA/D-DE for optimizing portfolios. We reviewed two different adaptations of each algorithm to enable the creation of optimal portfolios based on two different portfolio theories. All the work was done using real data from the Mexican Stock Market and the portfolios were evaluated according to: (1) historical performance; (2) future performance using unseen data; and (3) its comparison to well-known benchmarks. The three algorithms were able to produce highly competitive portfolios.

MOEA/D-DE seems to be the best algorithm in the research, the differential evolution operators are good enough to solve the problem. Additional research is required for more thorough parameter optimization, in that way we can guarantee the best results out the algorithm. Another enhancement to this algorithm is a convergence test, to know whether we have reached the limits of the search or not, instead of having an arbitrary number of generations.

CAPM together with the evolutionary algorithms was able to produce more accurate predictions of the future performance. However, it would be interesting in the future to see proposals that mix different portfolio theories with Evolutionary Algorithms. A more complete model could be done using a 3-objective optimization, taking the mean return, the standard deviation, and the Beta coefficient as separate objectives, MOEA/D-DE has proven to be the ideal candidate for such optimization.

The comparison between the EA portfolios and the benchmarks suggests that the Evolutionary Algorithms can produce portfolios that could be useful in real-life conditions, however the research fall short in representing life-like conditions. Additional investigations can be done taking into account transaction costs, taxes, and investment money to prove the performance in scenarios closer to real-life.

The advantages of using computer intelligence for financial applications are evident in this research. The MOEA/D-DE is a suitable algorithm for finding optimal portfolios. More research should be done to represent real-life conditions and that can be adapted to computer optimization. The current setting has given a confident expectation for Evolutionary Algorithms in portfolio optimization.

## 6. Bibliography

- [1] Markowitz, Harry (1958). *Portfolio Selection*. John Wiley & Sons. ISBN 978-1557861085.
- [2] Frank K. Reilly, Keith C. Brown (2012). *Investment Analysis and Portfolio Management*, 10th Edition. Cengage Learning. ISBN: 9780538482387.
- [3] A. Lambora, K. Gupta and K. Chopra, "Genetic Algorithm- A Literature Review," 2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon), Faridabad, India, 2019, pp. 380-384, doi: 10.1109/COMITCon.2019.8862255.
- [4] Q. Zhang and H. Li, "MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition," in *IEEE Transactions on Evolutionary Computation*, vol. 11, no. 6, pp. 712-731, Dec. 2007, doi: 10.1109/TEVC.2007.892759.
- [5] H. Zhang, Y. Zhao, F. Wang, A. Zhang, P. Yang and X. Shen, "A new evolutionary algorithm based on MOEA/D for portfolio optimization," *2018 Tenth International Conference on Advanced Computational Intelligence (ICACI)*, Xiamen, 2018, pp. 831-836, doi: 10.1109/ICACI.2018.8377569
- [6] Satchell, S., Scowcroft, A. A demystification of the Black–Litterman model: Managing quantitative and traditional portfolio construction. *J Asset Manag* **1**, 138–150 (2000). <https://doi.org/10.1057/palgrave.jam.2240011>
- [7] H. Li and Q. Zhang, "Multiobjective Optimization Problems With Complicated Pareto Sets, MOEA/D and NSGA-II," in *IEEE Transactions on Evolutionary Computation*, vol. 13, no. 2, pp. 284-302, April 2009, doi: 10.1109/TEVC.2008.925798.
- [8] P. Skolpadungket, K. Dahal and N. Harnpornchai, "Portfolio optimization using multi-objective genetic algorithms", 2007 IEEE Congress on Evolutionary Computation, Singapore, 2007, pp. 516-523, doi: 10.1109/CEC.2007.4424514.
- [9] S. Lim, M. Kim and C. W. Ahn, "A Genetic Algorithm (GA) Approach to the Portfolio Design Based on Market Movements and Asset Valuations," in *IEEE Access*, vol. 8, pp. 140234-140249, 2020, doi: 10.1109/ACCESS.2020.3013097.
- [10] Hoklie and L. R. Zuhail, "Resolving multi objective stock portfolio optimization problem using genetic algorithm," *2010 The 2nd International Conference on Computer and Automation Engineering (ICCAE)*, Singapore, 2010, pp. 40-44, doi: 10.1109/ICCAE.2010.5451372.
- [11] E. Zitzler and L. Thiele, "Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach," in *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 4, pp. 257-271, Nov. 1999, doi: 10.1109/4235.797969.

- [12] Gordon Y.N. Tang, "How efficient is naive portfolio diversification? an educational note" in *Omega, Elsevier* vol. 32, no. 2, pp. 155-160, April. 2004. <https://ideas.repec.org/a/eee/jomega/v32y2004i2p155-160.html>
- [13] Ishibuchi, H., Doi, K. & Nojima, Y. "On the effect of normalization in MOEA/D for multi-objective and many-objective optimization". *Complex Intell. Syst.* **3**, 279–294 (2017). <https://doi.org/10.1007/s40747-017-0061-9>
- [14] Y. Yuan, H. Xu, B. Wang, B. Zhang and X. Yao, "Balancing Convergence and Diversity in Decomposition-Based Many-Objective Optimizers," in *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 2, pp. 180-198, April 2016, doi: 10.1109/TEVC.2015.2443001.
- [15] W. Chen, R. Zhang, Y. Cai and F. Xu, "Particle Swarm Optimization for Constrained Portfolio Selection Problems," 2006 International Conference on Machine Learning and Cybernetics, Dalian, China, 2006, pp. 2425-2429, doi: 10.1109/ICMLC.2006.258773.
- [16] M. Chen, Jian Weng and Xia Li, "Multiobjective extremal optimization for portfolio optimization problem," 2009 IEEE International Conference on Intelligent Computing and Intelligent Systems, Shanghai, 2009, pp. 552-556, doi: 10.1109/ICICISYS.2009.5357781.
- [17] S. K. Mishra, G. Panda and S. Meher, "Multi-objective particle swarm optimization approach to portfolio optimization," 2009 World Congress on Nature & Biologically Inspired Computing (NaBIC), Coimbatore, 2009, pp. 1612-1615, doi: 10.1109/NABIC.2009.5393659.
- [18] Jianguo Cao and Liang Tao, "Improved particle swarm algorithm for portfolio optimization problem," 2010 The 2nd International Conference on Industrial Mechatronics and Automation, Wuhan, 2010, pp. 561-564, doi: 10.1109/ICINDMA.2010.5538246.
- [19] A. Talebi, M. A. Molaei and M. J. Sheikh, "Performance investigation and comparison of two evolutionary algorithms in portfolio optimization: Genetic and particle swarm optimization," 2010 2nd IEEE International Conference on Information and Financial Engineering, Chongqing, 2010, pp. 430-437, doi: 10.1109/ICIFE.2010.5609394.
- [20] G. A. V. Pai, "Multi-objective Differential Evolution Based Optimization of Risk Budgeted Global Asset Allocation Portfolios," 2014 2nd International Symposium on Computational and Business Intelligence, New Delhi, 2014, pp. 17-20, doi: 10.1109/ISCBI.2014.11.
- [21] R. Ramadhiani, M. Yan, G. F. Hertono and B. D. Handari, "Implementation of e-New Local Search based Multiobjective Optimization Algorithm and Multiobjective Co-variance based Artificial Bee Colony Algorithm in Stocks Portfolio Optimization Problem," 2018 2nd International Conference on Informatics and Computational Sciences (ICICoS), Semarang, Indonesia, 2018, pp. 1-6, doi: 10.1109/ICICOS.2018.8621646.
- [22] B. Y. Qu, Q. Zhou, J. M. Xiao, J. J. Liang, and P. N. Suganthan, "Large-Scale Portfolio Optimization Using Multiobjective Evolutionary Algorithms and Preselection Methods", *Mathematical Problems in Engineering*, Volume 2017, Article ID 4197914, 14 pages, doi: 10.1155/2017/4197914

[23] Divya Kumar, K.K.Mishra, "Portfolio optimization using novel co-variance guided Artificial Bee Colony algorithm", *Swarm and Evolutionary Computation*, Volume 33, April 2017, Pages 119-130, doi: 10.1016/j.swevo.2016.11.003

[24] H. Markowitz, W. F. Sharpe, and M. Miller, *Founders of Modern Finance: Their Prize Winning Concepts and 1990 Nobel Lectures*, CFA Institute, Charlottesville, Va, USA, 1991.

[25] T-J Chang, N Meade, J Beasley, Y Sharaiha. Heuristics for Cardinality Constrained Portfolio Optimization. *Computers and Operations Research* 27 (2000) 1271-1302.

[26] W. F. Sharpe. Capital Asset Prices: A Theory Of Market Equilibrium Under Conditions Of Risk. *Journal of Finance* 19 (1964), pp. 425–442, doi: 10.1111/j.1540-6261.1964.tb02865.x

[27] Storn, R., Price, K. Differential Evolution – A Simple and Efficient Heuristic for global Optimization over Continuous Spaces. *Journal of Global Optimization* **11**, 341–359 (1997). <https://doi.org/10.1023/A:1008202821328>

[28] Kwan C., Yang F., Chang C. (2007) A Differential Evolution Variant of NSGA II for Real World Multiobjective Optimization. In: Randall M., Abbass H.A., Wiles J. (eds) *Progress in Artificial Life. ACAL 2007. Lecture Notes in Computer Science*, vol 4828. Springer, Berlin, Heidelberg. [https://doi.org/10.1007/978-3-540-76931-6\\_30](https://doi.org/10.1007/978-3-540-76931-6_30)

# Appendix 1: Systematic Literature Review Table

Ref	Author	Year	Algorithm	Model	Metrics	Finding
15	W. Chen, R. Zhang, Y. Cai and F. Xu	2006	Particle Swarm Optimization	Constrained Mean-Variance + Transaction Costs	Average Fitness, Maximum Fitness	PSO with Mean-Variance model + Transaction costs and limits proved to be an effective method.
16	M. Chen, Jian Weng and Xia Li	2009	Multi-objective Extremal Optimization	Mean-Variance	Front Spread, Coverage	The results of MOEO were comparable to NSGA-II but not better.
7	H. Li and Q. Zhang	2009	MOEA/D	--	--	A multi-objective optimization algorithm that show great potential.
17	S. K. Mishra, G. Panda and S. Meher	2009	Multi-Objective Particle Swarm Optimization	Mean-Variance	Front Spread, Fitness	Multi-Objective Particle Swarm Optimization out-bested the rest of the algorithms in the study (NSGA-II, SPEA2 and PSFGA).
10	Hoklie and L. R. Zuhail	2010	Genetic Algorithm	Mean-Variance	Risk and Return	The results show that a simple genetic algorithm can provide good results for portfolio optimization and also that to find an optimal portfolio with GA, it is important to observe the different parameters of the GA.
18	Jianguo Cao and Liang Tao	2010	Improved Particle Swarm Optimization	Mean-Variance	Average Fitness	Proposed an Improved Particle Swarm Optimization (IPSO) and tested with Mean-Variance portfolio optimization, having better results than the original PSO.
19	A. Talebi, M. A. Molaeei and M. J. Sheikh	2010	Genetic Algorithm	Mean-Variance	The gap between predicted values and actual values in simulation.	The study showed that GA is more effective for portfolio optimization than PSO and that calculating yearly returns/variances are better than monthly.
20	G. A. V. Pai	2014	Multi-Objective Differential Evolution	Mean-Variance, Risk Budgeting Strategy	Sharpe ratio found vs Maximum Sharpe ratio	The algorithm was able to find the maximal Sharpe ration possible given the assets and results were consistent
23	Divya Kumar, K.K.Mishra	2017	Multi-objective Co-variance based Artificial Bee Colony	Mean-Variance	Generational Distance (from the Pareto front) and Spread	The result is good and can be worked further to accommodate more constraints.

22	B. Y. Qu, Q. Zhou, J. M. Xiao, J. J. Liang, and P. N. Suganthan	2017	MOEA/D, MODE, MODE-SS, MODE-NDS, NSGA-II and MOCLPSO	Mean-Variance	Maximum values in each dimension	In their experiment, the authors found that the normalized version of MOEA/D (NMOEA/D) performed better than the rest.
13	Ishibuchi, H., Doi, K. & Nojima, Y.	2017	MOEA/D	--	--	The advantages and use cases of normalizing objectives in MOEA/D.
21	R. Ramadhani, M. Yan, G. F. Hertono and B. D. Handari	2018	e-New Local Search based Multiobjective Optimization Algorithm	Constrained Mean-Variance	The gap between the solution and the efficient frontier	Results were compared to the efficient frontier given the set of assets. The e-NSLS appears to find better results than M-CABC.
5	H. Zhang, Y. Zhao, F. Wang, A. Zhang, P. Yang and X. Shen	2018	MOEA/D	Mean-Variance	The gap between the solution and the efficient frontier	The weight vector is a very important component for MOEA/D, they propose a new computing method for those vectors. The results shown in the paper are good.
9	S. Lim, M. Kim and C. W. Ahn	2020	Genetic Algorithm	Mean-Variance + CAPM	ROI in simulation	The algorithm attempts to find investment opportunities that can make a person rich, like AMZN, or NFLX, which in ten years created a profit for stock investors of more than 2500%. The algorithm failed to find such assets during the experimentation, however, results in general were positive.

## Appendix 2: Algorithm Configuration

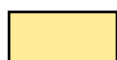
### Genetic Algorithm

Attribute	Value
Population Size	100
Number of Generations	500
Solution Size	473
Mutation Operator	Single-Point Mutation
Crossover Operator	Two-Point Crossover
Selection Operator	Tournament Selection
Tournament Size	10
Crossover Probability	0.9
Mutation Probability	0.9
Initialization Method	Original Algorithm <sup>†</sup>
Elitism	1

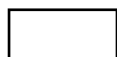
<sup>†</sup> Algorithm presented in section 3.2.1

### MOEA/D

Attribute	Value
Population Size	100
Number of Generations	500
Neighborhood Size	15
Solution Size	473
Mutation Operator	Single-Point Mutation
Crossover Operator	Two-Point Crossover
Normalization	1
Crossover Probability	0.3
Mutation Probability	0.9
Initialization Method	Random
Decomposition	Tchebycheff



Optimized parameters

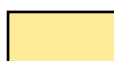
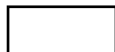


Arbitrary parameters

## MOEA/D-DE

Attribute	Value
Population Size	100
Number of Generations	500
Neighborhood Size	15
Solution Size	473
Mutation Operator	Polynomial Mutation
Crossover Operator	Differential Evolution
Normalization	1
Crossover Probability	0.5
Mutation Probability	0.7
Initialization Method	Original Algorithm <sup>†</sup>
Scaling Factor	0.5
Distribution Index	10
Decomposition	Tchebycheff

<sup>†</sup> Algorithm presented in section 3.2.1

	Optimized parameters
	Arbitrary parameters



# Appendix 3: Parameter Optimization Results

## Genetic Algorithm

### Baseline

Parameter	Value		
Crossover Probability	0.5		
Mutation Probability	0.5		
Elitism	1		
Initialization Method	Random		

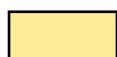
Average Fitness	Standard Deviation	Minimum	Maximum
166.5	11.2	143.3	184.7

### Crossover Probability

Crossover Rate	Average Fitness	Standard Deviation	Minimum	Maximum
0.1	159.0	17.1	121.4	181.8
0.3	164.2	14.1	135.6	179.5
0.5	166.5	11.2	143.3	184.7
0.7	167.2	9.3	146.4	181.2
0.9	168.6	7.0	157.2	178.0

### Mutation Probability

Mutation Rate	Average Fitness	Standard Deviation	Minimum	Maximum
0.1	112.4	13.6	91.5	135.2
0.3	151.9	21.8	107.9	175.0
0.5	164.0	11.3	137.2	178.7
0.7	170.8	8.1	159.2	182.3
0.9	175.6	14.1	151.1	194.2



Best value in each category

## Initialization Method

Initialization	Average Fitness	Standard Deviation	Minimum	Maximum
Random	169.6	12.3	151.0	188.3
Original Algorithm†	169.8	8.5	155.0	188.4

† Algorithm presented in section 3.2.1

## Elitism

Elitism	Average Fitness	Standard Deviation	Minimum	Maximum
0	163.8	9.7	145.7	182.5
1	166.5	10.6	146.4	179.7

## MOEA/D

### Baseline

Parameter	Value
Crossover Probability	0.5
Mutation Probability	0.5
Normalization	1
Initialization Method	Random
Neighborhood Size	10
Decomposition Method	Weighted Sum

Avg Num of Solutions	Std Dev	Average Max Fitness 1	Std Dev	Average Max Fitness 2	Std Dev
14.2	1.75119	0.75752	0.02845	-0.00243	0.00009

### Neighborhood Size

Neighborhood Size	Avg Num of Solutions	Std Dev	Average Max Fitness 1	Std Dev	Average Max Fitness 2	Std Dev
3	13.60000	2.06559	0.76468	0.03195	-0.00216	0.00013
8	13.60000	2.01108	0.73083	0.01623	-0.00232	0.00015
10	14.00000	3.23179	0.74303	0.02794	-0.00223	0.00021
15	14.60000	3.27278	0.72059	0.01518	-0.00210	0.00017
20	13.00000	2.44949	0.76575	0.03940	-0.00229	0.00018



Best value in each category

### Crossover Probability

Crossover P.	Avg Num of Solutions	Std Dev	Average Max Fitness 1	Std Dev	Average Max Fitness 2	Std Dev
0.1	11.7	2.90784	0.74496	0.02187	-0.00222	0.00017
0.3	13.6	3.27278	0.73025	0.02499	-0.00209	0.00016
0.5	13.2	3.25918	0.72681	0.01677	-0.00216	0.00018
0.7	13.1	1.96921	0.76252	0.03225	-0.00222	0.00017
0.9	13.5	2.75882	0.74383	0.02430	-0.00218	0.00016

### Mutation Probability

Mutation P.	Avg Num of Solutions	Std Dev	Average Max Fitness 1	Std Dev	Average Max Fitness 2	Std Dev
0.1	9.1	2.02485	0.70379	0.03414	-0.00256	0.00022
0.3	12.3	2.11082	0.70459	0.02084	-0.00237	0.00020
0.5	14.5	2.63523	0.76389	0.03786	-0.00225	0.00017
0.7	14.9	1.79196	0.76525	0.02584	-0.00215	0.00014
0.9	15.6	3.62706	0.75778	0.01722	-0.00203	0.00019

### Normalization

Normalize	Avg Num of Solutions	Std Dev	Average Max Fitness 1	Std Dev	Average Max Fitness 2	Std Dev
1	13.9	2.55821	0.73688	0.02536	-0.00219	0.00016
0	5.2	1.13529	0.77130	0.02052	-0.00362	0.00040

### Decomposition Method

Decomposition	Avg Num of Solutions	Std Dev	Average Max Fitness 1	Std Dev	Average Max Fitness 2	Std Dev
Weighted Sum	14.2	1.75119	0.75752	0.02845	-0.00243	0.00009
Tchevycheff	70	16.04854	0.75760	0.04482	-0.00253	0.00015



Best value in each category

## Initialization

Initialization	Avg Num of Solutions	Std Dev	Average Max Fitness 1	Std Dev	Average Max Fitness 2	Std Dev
Random	13.6	2.79682	0.74406	0.03345	-0.00216	0.00018
Original Algorithm†	13.1	3.17805	0.73833	0.01626	-0.00240	0.00007

† Algorithm presented in section 3.2.1

## MOEA/D-DE

### Baseline

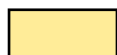
Parameter	Value
Crossover Probability	0.5
Mutation Probability	0.5
Normalization	1
Initialization Method	Random
Neighborhood Size	10
Decomposition Method	Weighted Sum
Distribution Index	1
Scaling Factor	1

Avg Num of Solutions	Std Dev	Average Max Fitness 1	Std Dev	Average Max Fitness 2	Std Dev
75	13.71131	0.76510	0.07915	-0.00386	0.00031

### Neighborhood Size

Neighborhood Size	Avg Num of Solutions	Std Dev	Average Max Fitness 1	Std Dev	Average Max Fitness 2	Std Dev
3	73.9	8.91254	0.77444	0.07060	-0.00339	0.00045
8	77.5	11.48187	0.77349	0.09294	-0.00377	0.00032
10	75	13.71131	0.76510	0.07915	-0.00386	0.00031
15	71.5	11.28667	0.77444	0.06704	-0.00349	0.00057
20	74.8	14.09334	0.77368	0.09078	-0.00365	0.00034



Best value in each category

### Crossover Probability

Crossover P.	Avg Num of Solutions	Std Dev	Average Max Fitness 1	Std Dev	Average Max Fitness 2	Std Dev
0.1	69.6	14.46221	0.77409	0.09868	-0.00331	0.00037
0.3	73.6	13.26817	0.76510	0.09118	-0.00356	0.00034
0.5	77.4	4.90351	0.76817	0.08527	-0.00386	0.00027
0.7	72.6	16.20151	0.77101	0.11361	-0.00362	0.00032
0.9	74.6	19.38613	0.77449	0.11553	-0.00369	0.00039

### Mutation Probability

Mutation P.	Avg Num of Solutions	Std Dev	Average Max Fitness 1	Std Dev	Average Max Fitness 2	Std Dev
0.1	67.2	13.13012	0.76255	0.10719	-0.00387	0.00031
0.3	79.5	11.90005	0.77518	0.11573	-0.00336	0.00040
0.5	74.5	14.00992	0.77449	0.11703	-0.00336	0.00039
0.7	75	12.77150	0.77253	0.08380	-0.00326	0.00047
0.9	69.7	9.45222	0.77426	0.06880	-0.00375	0.00042

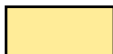
### Normalization

Normalization	Avg Num of Solutions	Std Dev	Average Max Fitness 1	Std Dev	Average Max Fitness 2	Std Dev
1	78.4	10.83410	0.75771	0.08120	-0.00335	0.00038
0	19.2	13.43131	0.77370	0.05922	-0.00365	0.00032

### Decomposition Method

Decomposition	Avg Num of Solutions	Std Dev	Average Max Fitness 1	Std Dev	Average Max Fitness 2	Std Dev
Weighted Sum	74.5	12.66886	0.77370	0.10470	-0.00366	0.00035
Tchebycheff	99.4	1.34990	0.76255	0.09079	-0.00363	0.00030

### Initialization

 Best value in each category

Initialization	Avg Num of Solutions	Std Dev	Average Max Fitness 1	Std Dev	Average Max Fitness 2	Std Dev
Random	64.5	15.00555	0.77102	0.09662	-0.00361	0.00035
Original Algorithm†	70.2	15.65461	0.76630	0.07289	-0.00373	0.00049

† Algorithm presented in section 3.2.1

### Scaling Factor

Scaling Factor	Avg Num of Solutions	Std Dev	Average Max Fitness 1	Std Dev	Average Max Fitness 2	Std Dev
0.4	70.9	20.04135	0.77520	0.07216	-0.003503	0.00027
0.5	82.6	9.57079	0.77448	0.04962	-0.00354	0.00037
0.75	78.1	16.61626	0.76476	0.09697	-0.00382	0.00017
0.9	78	13.59739	0.77518	0.08101	-0.003496	0.00031
1	78	12.98717	0.77504	0.02242	-0.00401	0.00024

### Distribution Index

Distribution Index	Avg Num of Solutions	Std Dev	Average Max Fitness 1	Std Dev	Average Max Fitness 2	Std Dev
1	75	15.52775	0.76544	0.10374	-0.00326	0.00036
5	68.6	19.06830	0.76609	0.05997	-0.00373	0.00029
10	79.1	7.50481	0.77440	0.08192	-0.00351	0.00040
20	72.9	12.20610	0.77441	0.01266	-0.00342	0.00042
50	76	11.13553	0.75563	0.12554	-0.00397	0.00030



Best value in each category

## Appendix 4: Results Summary

<b>GA + Mean-Variance</b>		<b>GA + CAPM</b>	
Training (2016, 2017, 2018)		Training (2016, 2017, 2018)	
Historical Mean Return	0.9447	Historical Mean Return	0.333395
Standard Deviation	0.3899	Standard Deviation	0.213736
Expected Return CAPM	0.0966	Expected Return CAPM	0.042508
Beta	0.6635	Beta	0.000099
Evaluation (2019)		Evaluation (2019)	
Return	0.2543	Return	0.2750
Standard Deviation	0.1985	Standard Deviation	0.1531
Beta	0.3447	Beta	0.0420
<b>MOEA/D + Mean-Variance (Low Risk)</b>		<b>MOEA/D + CAPM (Low Risk)</b>	
Training (2016, 2017, 2018)		Training (2016, 2017, 2018)	
Historical Mean Return	0.5971	Historical Mean Return	0.9230
Standard Deviation	0.1688	Standard Deviation	1.0433
Expected Return CAPM	0.0750	Expected Return CAPM	0.0429
Beta	0.3993	Beta	0.0048
Evaluation (2019)		Evaluation (2019)	
Return	0.1076	Return	0.3158
Standard Deviation	0.1023	Standard Deviation	0.1931
Beta	0.2591	Beta	0.4354
<b>MOEA/D + Mean-Variance (High Risk)</b>		<b>MOEA/D + CAPM (High Risk)</b>	
Training (2016, 2017, 2018)		Training (2016, 2017, 2018)	
Historical Mean Return	1.1138	Historical Mean Return	1.1613
Standard Deviation	0.5843	Standard Deviation	1.4320
Expected Return CAPM	0.0900	Expected Return CAPM	0.0498
Beta	0.5836	Beta	0.0900
Evaluation (2019)		Evaluation (2019)	
Return	0.3735	Return	0.4500
Standard Deviation	0.3043	Standard Deviation	0.2465
Beta	0.3547	Beta	0.4700
<b>MOEA/D-DE + Mean-Variance (Low Risk)</b>		<b>MOEA/D-DE + CAPM (Low Risk)</b>	
Training (2016, 2017, 2018)		Training (2016, 2017, 2018)	
Historical Mean Return	0.6132	Historical Mean Return	1.2552

Standard Deviation	0.2318	Standard Deviation	1.9361
Expected Return CAPM	0.1148	Expected Return CAPM	0.0476
Beta	0.8869	Beta	0.0622
Evaluation (2019)		Evaluation (2019)	
Return	0.1933	Return	0.4285
Standard Deviation	0.1775	Standard Deviation	0.2199
Beta	0.8154	Beta	0.5694
<b>MOEA/D-DE + Mean-Variance (High Risk)</b>		<b>MOEA/D-DE + CAPM (High Risk)</b>	
Training (2016, 2017, 2018)		Training (2016, 2017, 2018)	
Historical Mean Return	1.1400	Historical Mean Return	1.5388
Standard Deviation	0.7818	Standard Deviation	2.6463
Expected Return CAPM	0.0761	Expected Return CAPM	0.0749
Beta	0.4124	Beta	0.3974
Evaluation (2019)		Evaluation (2019)	
Return	0.5533	Return	0.5262
Standard Deviation	1.0632	Standard Deviation	0.2651
Beta	0.4543	Beta	0.8078



# Appendix 5: Portfolios Found

## **Genetic Algorithm + Mean-Variance**

ADVANCED MICRO DEVICES INC

23.05%

BOMBARDIER INC

17.42%

DIREXION SHARES ETF TRUST DAILY

23.05%

GOL LINHAS AEREAS INTELIGENTES

19.83%

VALE S.A.

16.62%

## **Genetic Algorithm + CAPM**

AMPHENOL CORP

8.16%

DRDGOLD LTD

14.37%

GRUPO FINANCIERO BANORTE

14.22%

GOL LINHAS AEREAS INTELIGENTES

8.92%

MONEX SAB DE CV

12.85%

NOVAGOLD RESOURCES INC

13.01%

GRUPO RADIO CENTRO SAB DE CV

10.74%

RH

10.89%

TERNIUM SA

6.80%

## **MOEA/D + Mean-Variance (Low Risk)**

AEROPORTS DE PARIS

7.98%

AUTODESK INC

9.17%

BOMBARDIER INC

9.05%

THE CHEMOURS COMPANY LLC

9.05%

DIREXION SHARES ETF TRUST DAILY

9.05%

GOL LINHAS AEREAS INTELIGENTES

7.03%

KERING

9.89%

NVIDIA CORP

7.03%

SONY CORP

11.32%

GRUPO TMM S.A.B.

11.32%

VALE S.A.

9.05%

**MOEA/D + Mean-Variance (High Risk)**

ADVANCED MICRO DEVICES INC

32.74%

DIREXION SHARES ETF TRUST DAILY

33.62%

GOL LINHAS AEREAS INTELIGENTES

33.62%

**MOEA/D + CAPM (Low Risk)**

APPLIED MATERIALS INC

28.57%

GOL LINHAS AEREAS INTELIGENTES

17.62%

PETROLEO BRASILEIRO SA PETROBRA

26.13%

PROMOTORA Y OPERADORA DE INFRST

27.65%

**MOEA/D + CAPM (High Risk)**

APPLIED MATERIALS INC

39.45%

GOL LINHAS AEREAS INTELIGENTES

24.36%

GRUPO KUO SAB DE CV

36.13%

**MOEA/D-DE + Mean-Variance (Low Risk)**

ADIDAS AG

11.19%

ADVANCED MICRO DEVICES INC

13.04%

AMAZON COM INC

16.03%

THE CHEMOURS COMPANY LLC

15.91%

GOLDGROUP MINING INC

5.35%

GOL LINHAS AEREAS INTELIGENTES

7.68%

KANSAS CITY SOUTHERN

16.54%

VALE S.A.

14.22%

**MOEA/D-DE + Mean-Variance (High Risk)**

ADVANCED MICRO DEVICES INC

31.08%

THE CHEMOURS COMPANY LLC

20.18%

GOL LINHAS AEREAS INTELIGENTES

48.72%

**MOEA/D-DE + CAPM (Low Risk)**

APPLIED MATERIALS INC

54.99%

CIA ENERGETICA MINAS GERAIS-CEM

45.01%

**MOEA/D-DE + CAPM (High Risk)**

APPLIED MATERIALS INC

76.08%

CIA ENERGETICA MINAS GERAIS-CEM

23.91%



