



Università
Ca' Foscari
Venezia

**Master's Degree Programme
Second Cycle (D.M. 270/2004)
in Economics and Finance – Finance**

Final Thesis

**GAs and PSO:
two metaheuristic methods
for portfolio optimization**

Supervisor:

Prof. Marco Corazza

Graduand:

Cristina Moret

839849

Academic Year

2017/2018

Contents

Introduction	iv
1 Portfolio Selection Theory	3
1.1 Markowitz Model	3
1.2 Assumptions	4
1.3 Portfolio Selection Model	4
1.3.1 Measure of Risk and Return	5
1.3.2 Mean-Variance Dominance Criterion	6
1.3.3 Portfolio Selection	8
1.4 Criticisms to Markowitz Model	9
1.5 Improvements of the Markowitz Model	10
1.5.1 Coherent Risk Measures	13
1.5.2 Two-Sided Coherent Risk Measure	17
2 Genetic Algorithms and Particle Swarm Optimization	19
2.1 Metaheuristics	19
2.1.1 History Overview	21
2.2 Genetic Algorithms	22
2.2.1 A simple GA	25
2.2.2 How do GAs work	26
2.2.3 The Schema Theorem	33
2.3 Particle Swarm Optimization	37
2.3.1 Swarm Intelligence	37
2.3.2 How does a PSO algorithm works	40
2.3.3 Parameters of the Algorithm	42
2.3.4 Modifications of the PSO	43
2.3.5 Population Topology	46
2.4 Comparison between GA and PSO	48

3 Risk Measure and Portfolio Selection Model	51
3.1 Two-sided coherent risk measure in detail	51
3.2 Portfolio Selection Model	53
4 Applications to the FTSE MIB Stock Index	61
4.1 Problem Setting	61
4.2 Application and discussion	65
Conclusions	80
Bibliography	83

Introduction

The main theme of this work is the portfolio selection problem with the use of metaheuristics. Portfolio management is part of Modern Portfolio Theory, a subject introduced by Harry Markowitz in the 1950s. In Chapter 1 it is described the model proposed by Markowitz, that is known as Mean-Variance portfolio selection model. This model has been revolutionary when proposed, with the years passing and the evolution of the markets, though, this model faced some limitations. The most important limitations that arised are the fact that the assumptions on which this model is based do not reflect the real world, and the risk measure adopted. As the name suggests, the risk measure adopted by Markowitz is the variance. As portfolio theory became a more and more studied subject, it has been realized that the variance is not a suitable risk measure for financial portfolios. The concept of coherent risk measure has been developed and a new class of measures of risk has been created. In particular a new two-sided risk measure has been introduced in 2008 by Chen and Wang [15].

Chapter 2 focuses on metaheuristics. Metaheuristics are a way to find higher level solutions by trial and error for a complex problem in a reasonable time. The main advantage of using metaheuristics, in fact, is the limited time needed for computations. The solutions provided by metaheuristics are in general good solutions, even if they may not be the optimal one. This work will focus in particular on two bio-inspired metaheuristics Genetic Algorithms and Particle Swarm Optimization, that will then be applied to a portfolio selection problem.

The other limitations faced by the Markowitz Mean-Variance portfolio selection model mentioned above are the assumptions on which it is based. An example of an unrealistic assumption is the one of frictionless markets. This means that the portfolio selected with Markowitz's model assumes that markets operate with no transaction costs, no taxes and that an investor can buy or sell any quantity of assets he desires. In order to make the analysis more realistic, a portfolio that takes all these facts into consideration has been choosen. In Chapter 3, the realistic portfolio proposed by Corazza, Fasano and Gusso in [16] is described. By using this portfolio and the two-sided risk measure, the optimization problem to solve

is a non-linear and mixed-integer constrained problem, that makes it a NP-hard problem. To make the problem easier to solve, an exact penalty method is applied. The latter transforms the problem from a constrained one to an unconstrained problem.

In Chapter 4 the two metaheuristic methods described in Chapter 2 are used to solve the portfolio selection problem described in Chapter 3. The optimization problem is applied to the FTSE MIB index and the data used are the daily closing prices from April 2012 until March 2018. The data have been divided in eight periods which in turn are divided in 6 months of in-sample and 3 months of out-of-sample period. In particular, the monotonicity property of the risk measure and the out-of-sample portfolio performances are analyzed. To conclude this work, a comparison between the results provided by PSO and GAs is made.

Chapter 1

Portfolio Selection Theory

Portfolio selection is a very significant topic in economics and finance. The most outstanding work in this matter was made by Harry Markowitz, an American economist, and the related field of research is known as Modern Portfolio Theory. In this chapter, I will briefly describe the portfolio selection model introduced by Markowitz and analyze its advantages and limitations in order to propose some improvements.

1.1 Markowitz Model

Markowitz published his paper “Portfolio Selection” in 1952 and is now acknowledged as the basis of Modern Portfolio Theory. Markowitz won the Nobel Prize in Economic Sciences in 1990. Markowitz model is known as the Mean-Variance portfolio selection model because it is based on the expected returns (mean) and the standard deviation (variance) of the portfolio, he said in [1]:

“ We consider the rule that the investor does (or should) consider expected return a desirable thing and variance of return an undesirable thing”.

The level of risk of the portfolio is given by the variance of the assets in the portfolio, this model aims to find assets that collectively have a lower variance than they have singularly. Given the level of risk of the assets in the portfolio, Markowitz showed how to obtain the maximum return combining different financial instruments in the same portfolio, so developing a theory to optimize investments’ returns thanks to the creation of a diversified portfolio. In the Markowitz theory there is a return-risk trade-off since the investor wants to both maximize return while minimizing risk.

1.2 Assumptions

In this section I am going to describe the assumptions on which the Mean-Variance portfolio selection model is based on:

- Markets are frictionless, this means that there are no transaction costs, no taxation, and the individual assets are indefinitely divisible, i.e. the investor can buy or sell any quantity he desires;
- Investors are price-taker, their actions can not affect the probability distribution of returns on the available securities;
- Each investment opportunity is measured over the same holding period and is represented by a probability distribution of returns;
- Investors' aim is to maximize the rate of return they can obtain from an investment;
- Investors are rational and risk averse. That means that they are completely aware of the risk contained in an investment and take positions based on the risk, demanding a higher return for accepting higher risk.

1.3 Portfolio Selection Model

Now I am going to describe the portfolio selection process. As Markowitz said in [2]: "A portfolio analysis starts with information concerning individual securities. It ends with conclusions concerning portfolios as a whole".

The selection process of a portfolio can be divided into three stages:

1. Identify a tool to measure the uncertain situation associated to a given investment choice; that means to find an appropriate measure of risk and return.
2. Define an efficiency criterion by which to divide all possible investment choices into two mutually exclusive sets: an efficient set and an inefficient set.
3. Select the optimal portfolio for the investor among the efficient ones, taking into consideration the investor's taste for risk.

1.3.1 Measure of Risk and Return

The tools to measure the uncertain situation associated to the investment choice are:

- The mean of the single-period rate of return. From a financial point of view, the mean of the rate of return of an investment choice is a measure of the profitability of the investment choice itself, that is desirable.
- The variance of the single-period rate of return. From a financial point of view, it is a measure of the risk of the investment choice and it is undesirable.

To define the expected value and the variance for individual securities, let X be a discrete random variable $X = (x_1, p_1), \dots, (x_i, p_i), \dots, (x_M, p_M)$ where x_i with $i = 1, \dots, M$, is the i -th realization of X , the possible return from a given asset, and p_i , with $i = 1, \dots, M$, is the probability of occurrence of x_i , with $0 \leq p_i \leq 1$ for all i and $\sum_{i=1}^M p_i = 1$. Then:

$$E(X) = \sum_{i=1}^M x_i p_i$$

$$Var(X) = \sum_{i=1}^M (x_i - E(x))^2 p_i$$

If X is a continuous random variable characterized by a cumulative probability distribution function $F(\cdot)$ and/or by a probability density function $f(\cdot)$, then the expected value and the variance for an individual security can be expressed as follows:

$$E(X) = \int_{-\infty}^{+\infty} t dF(t) \quad \text{and/or} \quad E(X) = \int_{-\infty}^{+\infty} t f(t) dt$$

$$Var(X) = \int_{-\infty}^{+\infty} (t - E(X))^2 dF(t) \quad \text{and/or} \quad Var(X) = \int_{-\infty}^{+\infty} (t - E(X))^2 f(t) dt$$

Since a portfolio can be defined as the weighted average of individual securities, we can now define the expected rate of return and the variance of the portfolio. We first define the rate of return of the portfolio:

$$R_P = x_1 R_1 + \dots + x_N R_N = \sum_{i=1}^N x_i R_i$$

where R_i is a random variable representing the return of the i -th asset and x_i the percentage of capital invested in the same i -th asset. Let r_i and σ_i^2 be respectively the expected rate of return and the variance of the i -th asset, with $i = 1, 2, \dots, N$. Then the expected value and variance of the portfolio will be:

$$E(R_P) = \sum_{i=1}^N x_i r_i =: r_P$$

$$Var(R_P) = \sum_{i=1}^N x_i^2 \sigma_i^2 + 2 \sum_{i=1}^N \sum_{j=i+1}^N x_i x_j \sigma_{i,j} = \sum_{i=1}^N x_i^2 \sigma_i^2 + 2 \sum_{i=1}^N \sum_{j=i+1}^N x_i x_j \rho_{i,j} \sigma_i \sigma_j =: \sigma_P^2$$

where $\sigma_{i,j} = \rho_{i,j} \sigma_i \sigma_j$ is the covariance and $\rho_{i,j} = [-1, 1]$ is the Bravais-Pearson linear correlation coefficient between R_i and R_j . The mean and the variance of the portfolio can also be expressed using the vectorial notation:

- $r_P = \mathbf{x}'\mathbf{r}$;
- $\sigma_P^2 = \mathbf{x}'\mathbf{V}\mathbf{x}$ where \mathbf{V} is the usual variance-covariance matrix.

1.3.2 Mean-Variance Dominance Criterion

The efficiency criterion proposed by Markowitz is based on the concepts of mean and variance. He argued in [3] that the minimization of variance given a level of expected return, lead to the “right kind” of diversification for the “right reason”.

Let X and Y be two random variables with mean equal to μ_X and μ_Y respectively, and variance equal to σ_X^2 and σ_Y^2 , then we say that X dominates Y , or that X is preferred to Y , in the mean-variance dominance sense if:

- $\mu_X \geq \mu_Y$
- $\sigma_X^2 \leq \sigma_Y^2$

and at least one of these two inequalities is satisfied in the strong form. A portfolio is efficient when it is not dominated by other portfolios. This means that there are no portfolios with greater expected return and lower variance than the efficient portfolio.

The portfolio selection problem consists in finding the portfolio that minimizes the variance, among all possible portfolios with the same rate of return, given that you invested all the capital. The basic version of the portfolio selection problem in the case of N assets can be stated as follows:

$$\begin{aligned} & \text{minimize } x_1, \dots, x_N \quad \mathbf{x}'\mathbf{V}\mathbf{x} \\ & \text{subject to } \begin{cases} \mathbf{x}'\mathbf{r} = \pi \\ \mathbf{x}'\mathbf{e} = 1 \end{cases} \end{aligned}$$

where:

- π is the expected rate of return that the investor wishes the portfolio realizes;
- $\mathbf{x} = (x_1, \dots, x_N)$ is the N-column vector of the proportion of wealth invested in the i -th asset of the portfolio with $i = 1, 2, \dots, N$;
- \mathbf{V} is the N x N matrix of covariances between the assets;
- $\mathbf{r} = r_1, \dots, r_N$ is the N-column vector of mean returns of the assets;
- \mathbf{e} is an N-column vector composed by ones;

The portfolio selection problem has unique solution and is given by the following theorem:

Let \mathbf{V} be a NxN-matrix of variances and covariances and let \mathbf{r} be the N-vector of mean returns. If \mathbf{V} is nonsingular and positive definite and if $r_i \neq r_j$ for some $i, j = 1, \dots, N$ then the portfolio selection problem has the following unique solution:

$$x^* = \frac{(\gamma\mathbf{V}^{-1}\mathbf{r} - \beta\mathbf{V}^{-1}\mathbf{e})\pi + (\alpha\mathbf{V}^{-1}\mathbf{e} - \beta\mathbf{V}^{-1}\mathbf{r})}{\alpha\gamma - \beta^2}$$

where:

- $\alpha = \mathbf{r}'\mathbf{V}^{-1}\mathbf{r}$
- $\beta = \mathbf{r}'\mathbf{V}^{-1}\mathbf{e} = \mathbf{e}'\mathbf{V}^{-1}\mathbf{r}$
- $\gamma = \mathbf{e}'\mathbf{V}^{-1}\mathbf{e}$

We can interpret the financial meaning of the assumptions of the theorem as follows:

- \mathbf{V} has to be nonsingular, this means that $\det(\mathbf{V}) \neq 0$. When a matrix is singular it means that there is linear combination among columns or rows. Any column or row of the variance-covariance matrix contains the risk structure of an asset. We don't want the risk structure of an asset to be defined by the linear combination of other assets' risk structures, for this reason the determinant has to be different from zero.
- \mathbf{V} is positive definite: all the assets that we are considering are really risky, all the variances are positive.

- $r_i \neq r_j$ for some $i, j = 1, \dots, N$: there has to be at least one asset with a different expected rate of return, to avoid investing all the capital in the riskless asset.

The composition of the efficient frontier depends on: \mathbf{V} , \mathbf{r} and μ . We have that:

- $E(R_{P^*}) = r_{P^*} = \mathbf{x}'\mathbf{r} = \mu$;
- $Var(R_{P^*}) = \sigma_{P^*}^2 = \frac{\gamma\pi^2 - 2\beta\pi + \alpha}{\alpha\gamma - \beta^2}$, which describes a parabola in the variance-mean plane;
- $StDev(R_{P^*}) = \sigma_{P^*} = \left(\frac{\gamma\pi^2 - 2\beta\pi + \alpha}{\alpha\gamma - \beta^2}\right)^{\frac{1}{2}}$, which describes a hyperbola in the standard deviation-mean plane.

1.3.3 Portfolio Selection

In the third step, we have to select the optimal portfolio for the investor. In order to do this we have to take into consideration the investor's taste for risk. Two objectives are common to all investors:

- They prefer returns to be high;
- They are risk-averse: they prefer the return to be stable, not subject to uncertainty.

Now the investor has to decide, among different portfolios, the one that produces the greatest wealth at the end of the period. This can be done with the utility function.

A utility function is a function U defined on real numbers that represent possible wealth levels, giving real values. Using the utility function, it is possible to rank all alternative future wealth levels, by evaluating their expected utility values. In practice, one has to compare the values of $E[U(x)]$ and $E[U(y)]$ where x and y are two outcome random wealth variables, and the highest expected value is preferred. If x and y are real values with $x > y$, then $U(x) > U(y)$, this means that the utility function is an increasing continuous function. Furthermore, the utility function has to be concave¹ in order to represent the risk aversion of the investor.

¹Concave utility and risk aversion: a function defined on an interval $[a, b]$ of real numbers is said to be concave if for any α with $0 \leq \alpha \leq 1$ and any x and y in $[a, b]$ there holds:

$$U[\alpha x + (1 - \alpha)y] \geq \alpha U(x) + (1 - \alpha)U(y)$$

A utility function U is said to be risk averse on $[a, b]$ if it is concave on $[a, b]$. If U is concave everywhere, it is said to be risk averse.

An individual utility function can be measured asking individual investors to assign values to various risky alternatives. The expected utility approach is consistent with the Markowitz portfolio selection problem in one of the following ways:

- Using a quadratic utility function, or
- Making the assumption that the random variables that characterize returns are Gaussian random variables.

The quadratic utility function used by Markowitz is the following:

$$U(R_P) = R_P - \frac{a}{2}R_P^2$$

where:

- R_P is a random variable of the return of the portfolio;
- a is a coefficient that reflects the risk aversion of the investor. a must be strictly positive and the higher its value, the higher the risk aversion.

The optimal portfolio is selected among the ones belonging to the efficient frontier and is the portfolio that maximizes the expected utility of the investor.

1.4 Criticisms to Markowitz Model

Even if the Markowitz portfolio selection problem is the fundamental of Modern Portfolio Theory, it is not perfect. Through the years some problems with the mean-variance approach arised and criticism have been made. In this section I am going to present the major criticisms to the Markowitz model.

First of all, some assumptions of the model are not realistic [5]: real world markets are not frictionless, transaction costs and taxes are present and have to be taken into account. Moreover assets are not indefinitely divisible, investors can not buy security of any size because some securities have a minimum order size and it is not possible to buy or sell securities in fractions. Other unrealistic assumptions are that investors are always acting rationally and that the assets' returns are independent.

Most of the time investors do not have an exact idea of the potential returns, usually their expectations are unbiased. The determination of the optimal rate of return of the portfolio assumes that the world is stationary, which is not in reality. It also

assumes that investors are conscious of using the quadratic utility function and that they are capable to quantify their propensity to risk.

Markowitz model is based on the idea of diversification: the variance of the return of the portfolio can be reduced by including additional assets in the portfolio. This effect has its limits, it tends to decrease as additional assets are included in the portfolio. Increasing the number of assets means that the size of the variance-covariance matrix increases as well and this can make more complicated the portfolio selection problem.

Another criticism is that the model does not account for asymmetries and/or kurtosises in return distributions. An assumption of the model, if the investor wants to use a generic utility function and not the quadratic one, is that returns are normally distributed. This is not the case with financial instruments: they tend to have distributions that are generally skewed and with fat tails.

It has also been argued that the Markowitz model is based on two conflicting optimization criteria: the minimization of the risk (variance) and the maximization of returns. These two principles are conflicting because in general higher returns lead to higher risk, while lower risk leads to lower returns.

In conclusion, the use of the variance as measure of risk has been largely questioned. The variance is a symmetric risk measure that takes into account the down-side as well as the up-side of risk, giving them equal weights. This means that also the positive variance is considered risky, that it is not risky for investors. Investors treat differently positive and negative variance. This lead to the fact that the variance of the rate of return of an investment choice is not a good measure of the risk of the investment choice itself.

1.5 Improvements of the Markowitz Model

The portfolio selection problem proposed by Markowitz can be improved in two ways.

The first way to is to improve the system of constraints. Changing and adding constraints is useful in order to address the problem of the unrealistic assumptions of the Markowitz model. In this way, it is possible to take into account market frictions, such as transaction costs, and the infinite divisibility of the assets. Some important categories of the constraints on the divisibility of assets are:

- restrictions related to the minimum number of lots that have to be bought or sold only in integer numbers, not in fractions;

- restrictions related to the maximum positive integer number of different assets that can be bought or sold;
- restrictions related to the minimum positive integer number of minimum transaction lots of an asset that has to be bought or sold.

The introduction of these constraints, called mixed-integer constraints, has at least two implications. The first one is related to the fact that checking the eligibility of the system of constraints of this mathematical programming problem is, in general, a NP-complete² problem. The second one is related to the fact that solving these mathematical programming problems is, in general, a NP-hard³ problem.

The second way to improve the Markowitz portfolio selection model consists in working on the objective function. Following I will consider different measures of risk. Knowing the drawbacks of the variance, Markowitz itself proposed an alternative measure of risk of the rate of return of an investment choice: the semi-variance.

$$semi - Var(R) = \sum_{i=1}^M (\min\{0, x_i - E(R)\})^2 p_i$$

This formula is valid when R is a discrete random rate of return. If R is a continuous random variable the semi-variance is defined as follows:

$$semi - Var(R) = \int_{-\infty}^{+\infty} (\min\{0, t - E(R)\})^2 dF_R(t)$$

and/or

$$semi - Var(R) = \int_{-\infty}^{+\infty} (\min\{0, t - E(R)\})^2 f_R(t) d(t)$$

With the use of the semi-variance are taken into account only the performances below the mean, the downside risk. The positive variance is no longer considered as risk but as upside potential, since it represents performances of the portfolio above the expectations and any rational investor does not see them as risk but as opportunities.

As Markowitz stated in [2]: "analyses based on the semi-variance tend to produce better portfolios than those based on the variance". This is because the analysis based on the semi-variance concentrates on reducing losses, while the analysis

²NP-Complete are problems for which no fast solution is known. The time required to solve these problems increases very quickly as the size of the problem grows.

³NP-hard are problems that are at least as hard as NP-complete problems.

based on the variance considers both extremely high and extremely low returns equally undesirable, and seeks to eliminate both of them. Nonetheless, the use of semi-variance does not solve all the problems that arise with the variance and has itself drawbacks. To derive the efficient set based on the semi-variance requires much more calculations than deriving it based on the variance.

Now I am going to explain another measure of risk that has been taken into consideration, in place of the variance, for the portfolio selection problem.

The need for very sophisticated risk control models, required by financial institutions, increased the necessity for new risk measures. The concept of Value at Risk (VaR) was introduced in 1994 and has been approved by the Basel Committee to calculate capital reserve needed to cover market risks. The classical measures of risk used before are statistical indices for variability, not borne for financial risk. Value at risk is considered a modern measure that does not tell you the maximum loss that you can have, it tells you the minimum you can lose. VaR answers the following questions: how much can you expect to lose at minimum in a given time horizon with a given probability? What is the percentage of the value of the investment that is at risk? VaR is defined as the minimum level of loss at a given confidence level ϵ for a predefined time horizon. The confidence levels normally used are 95% and 99%. Denote by X the risky payoff and by $(1 - \epsilon)$ the confidence level with $0 \leq \epsilon \leq 1$, then formally:

$$VaR_\epsilon(X) = -\inf_x \{x | P(X \leq x) \geq \epsilon\}$$

The value at risk has two properties that does not apply to the variance:

- Denote by C a riskless payoff, then:

$$VaR_\epsilon(X + C) = VaR_\epsilon(X) - C$$

This property is called transitional invariance.

- Denote by λ a positive constant, then

$$VaR_\epsilon(\lambda X) = \lambda VaR_\epsilon(X)$$

This property is the positive homogeneity property.

This measure of risk has been widely used because it gives a compact representation of the risk level and it measures only the downside risk.

Nevertheless, VaR has also some drawbacks. As Szegö said in [10]: “Unfortunately, just like Markowitz approach, with the explicit encouragement of

regulators, VaR has become another “solution in search of a problem” and was wrongly adopted as a risk measure.”

The measure of risk used for selecting the optimal portfolio has to be coherent. As it will be explained in detail in the next section, a measure of risk is defined coherent when it satisfies four properties: positive homogeneity, sub-additivity, monotonicity and transitional invariance. The Value at Risk does not satisfy always the sub-additivity property, in particular the reasonable diversification effect is not always present. That is, denoting by X and Y two risky payoff:

$$VaR_\epsilon(X + Y) > VaR_\epsilon(X) + VaR_\epsilon(Y)$$

This means that in some situations, the VaR of the portfolio can be higher than the summation of the VaR of the single assets. Another drawback is that the VaR does not tell you the maximum loss but it indicates the minimum level of loss, as a consequence the loss can be higher than the VaR. The lack of the sub-additivity property, leads to the lack of convexity ⁴ and makes VaR unsuitable to measure risk in real life portfolios. Convexity is an important characteristic since it is necessary to reward diversification, which is a key feature in optimizing the portfolio. There is only one special case when the sub-additivity property is satisfied and it is when the joint distribution of return is elliptic.

1.5.1 Coherent Risk Measures

In order to find a proper objective function, a new class of measures of risk has been developed: the coherent risk measures. They are characterized by a set of properties which are considered useful in measuring financial risk. A measure of risk has been defined by Szegő as “establishing a correspondence ρ between a space X of a random variable and a nonnegative real number, $\rho : X \rightarrow R$.”

Such a scalar measure of risk allows to compare and order different investments on the basis of their risk value. A coherent risk measure has to satisfy four properties: positive homogeneity, sub-additivity, monotonicity and transitional invariance.

(a) *Positive homogeneity:*

$$\rho(\lambda X_i) = \lambda \rho(X_i)$$

⁴Convexity: for any $\lambda \in (0, 1)$ and any risky positions $X_1, X_2 \in \mathcal{X}$ a risk measure is said to be coherent when:

$$\rho(\lambda X_1 + (1 - \lambda)X_2) \leq \lambda \rho(X_1) + (1 - \lambda)\rho(X_2)$$

for all positive real number λ and for all X_i , with $i = 1, \dots, N$. This means that the size of the investment has to affect linearly the size of the risk.

(b) *Sub-additivity*:

$$\rho(X_i + X_j) \leq \rho(X_i) + \rho(X_j)$$

for all X_i and X_j , with $i, j = 1, \dots, N$. This means that the risk measure has to be able to perform risk contraction. Economically speaking it means that adding a risky investment choice to another risky investment choice allows to decrease the overall risk of the underlying investment.

(c) *Monotonicity*:

$$X_i \preceq X_j \Rightarrow \rho(X_i) \geq \rho(X_j)$$

for all couple of X_i and X_j , with $i, j = 1, \dots, N$. This means that the risk measure has to reflect the investor preference of asset.

(d) *Transitional invariance*:

$$\rho(X_i + X_{N+1}) = \rho(X_i) - \alpha$$

for all riskless portfolio X_{N+1} having return equal to a real number α , and for all X_i , with $i = 1, \dots, N$. This means that adding a riskless investment choice to a risky one, the risk of the overall underlying investment decreases with respect to the risk of the risky investment choice.

Any ρ that satisfies property (a) and (b), provides a convex measure of risk, while a measure that satisfies all four properties is said to be coherent.

There are several measures of risk that are coherent, now I will explain some of them:

- **Expected Regret**

$$G_\alpha(X) = \int_{y \in R^m} [f(x, y) - \alpha]^+ p(y) dy$$

with $[u]^+ = \max\{0, u\}$. The expected regret is the expected value of the distribution beyond the threshold α . This measure can be calculated using a linear programming model based on a scenario approach: $\min \mathbf{p}^T [\mathbf{y} - \alpha]^+$. The objective function is $\mathbf{p}^T [\mathbf{y} - \alpha]^+$, where \mathbf{p} is the probability associated at each scenario and \mathbf{y} the portfolio losses exceeding α for each scenario j . The objective function is the weighted average of the probability of each scenario

of the portfolio regrets and it represents the mean with respect to all possible scenarios of the portfolio losses higher than the threshold α .

- **Conditional Value at Risk**

CVaR is a coherent risk measure that has several advantages over the Value at Risk. In particular, it is able to quantify the losses that might occur in the tail and it can be expressed by a minimization formula that can be incorporated in minimization problems. The CVaR is defined as the expected loss at a given confidence level α given that the loss is greater than the VaR at that level. Mathematically, the α -CVaR of the loss associated with a decision x is the value $\phi_\alpha(x) = \text{mean of all the } \alpha\text{-tail distribution of } z = f(x, y)$, where the distribution in question is the one with distribution function $\Psi_\alpha(x, \cdot)$ defined by:

$$\Psi_\alpha(x, \zeta) = \begin{cases} 0 & \text{for } \zeta < \zeta_\alpha(x) \\ [\Psi(x, \zeta) - \alpha]/[1 - \alpha] & \text{for } \zeta \geq \zeta_\alpha(x) \end{cases}$$

where $\zeta_\alpha(x)$ is equal to VaR_α associated to portfolio x . Note that $\Psi_\alpha(x, \cdot)$ is a non-decreasing and right-continuous distribution function, with $\Psi_\alpha(x, \zeta) \rightarrow 1$ as $\zeta \rightarrow \infty$. This means that the α -tail distribution $\phi_\alpha(x)$ is well defined through $\Psi_\alpha(x, \zeta)$.

- **Tail Conditional Expectation**

In order to find an answer to the question: "What is the expected loss incurred in the α worst cases of our portfolio?" where $\alpha \in (0, 1)$ is a percentage which represents a sample of "worst cases" for the portfolio we want to analyze, the concept of Tail Conditional Expectation has been developed. TCE is given by the conditional expected value below the quantile:

$$TCE^{(\alpha)}(X) = -E\{X|X \leq x^{(\alpha)}\}$$

where X is a random variable describing the future value of a profit or loss, in a fixed time horizon T , of a portfolio and $x^{(\alpha)}$ is the related quantile of the distribution. The formula holds for distribution functions of the portfolio that are continuous. For other general distributions TCE does not answer the question, because the event $\{X \leq x^{(\alpha)}\}$ may have a greater probability than α , thus it may be greater than the set of selected "worst cases". As a consequence, with general distributions the Tail Conditional Expectation may violate the sub-additivity assumption and therefore be no longer coherent. In the case

of continuous distribution functions, on the other hand, the sub-additivity property is not violated and TCE is a coherent measure of risk.

- **Worst Conditional Expectation**

Worst Conditional Expectation, on the contrary of TCE, always satisfy the sub-additivity assumption. Mathematically it is expressed as follows:

$$WCE_\alpha(X) = -\inf\{E[X|A] | P[A] > \alpha\}$$

where X is a real-valued random value in a probability space (Ω, \mathcal{A}, P) that represents the loss (or profit) of the portfolio and $A \in \mathcal{A}$ is an event. $WCE_\alpha(X)$ does not depend only on the probability distribution of X , but also on the structure of the underlying probability space. It has been proved in Artzner et al. (1999) that the following inequality holds: $WCE_\alpha \geq TCE^\alpha$. Both risk measures, TCE and WCE, were developed in order to answer the question "how bad is bad" and have first identified the retention level with the quantile that is used in the financial risk measurement area. Notice that TCE may not always be coherent but it suits well for practical applications, while WCE is always coherent but it is useful only in theoretical setting since it requires the knowledge of the whole underlying probability space.

- **Expected Shortfall**

The Expected Shortfall tries to put together the positive aspects of TCE and WCE. ES represents the average loss in $(1 - \alpha)\%$ of the worst cases. Acerbi et al. (2002) defines the Expected Shortfall as follows:

"Let X be the profit-loss of a portfolio on a specified time horizon T and let $\alpha = A\% \in (0, 1)$ some specified probability level. The expected $A\%$ shortfall of the portfolio is then defined as

$$ES^{(\alpha)}(X) = -\frac{1}{\alpha} \left(E[X \mathbf{1}_{\{X \leq x^{(\alpha)}\}}] - x^{(\alpha)}(P[X \leq x^{(\alpha)}] - \alpha) \right),$$

where $x^{(\alpha)} = VaR$. The term $x^{(\alpha)}(P[X \leq x^{(\alpha)}] - \alpha)$ represents the exceeding part that has to be subtracted from the expected value when $\{X \leq x^{(\alpha)}\}$ has a probability larger than $\alpha = A\%$. In the particular case when $P[X \leq x^{(\alpha)}] = \alpha$, the probability distribution is continuous and we have that $ES^{(\alpha)} = TCE^{(\alpha)}$.

The Expected Shortfall is always a coherent risk measure that leads to a unique solution of the problem. When the random variables are continuous, the definition of ES coincides with the one of CVaR.

1.5.2 Two-Sided Coherent Risk Measure

The coherent risk measures so far described share a common disadvantage: they are one-sided. This means that they take into consideration only the lower part of the distribution information, i.e. the part of the losses. This does not reflect the real world since an investor, when he wants to buy a stock, has to take into consideration the actions of the seller of the stock, meaning that potential losses from both demand and offer sides have to be taken into account simultaneously. Moreover, one-sided coherent risk measures tried to solve the problem affecting VaR, they took into consideration also losses beyond the Value at Risk threshold, but they considered only the linear probability weighted combination of losses beyond this threshold, without taking into account different orders of moments of a non-normal distribution. Another limitation of this coherent risk measures is that a real application of them is not proposed, most of them were proposed theoretically but were not applied to solve realistic portfolio selection problems.

In order to overcome these limitations, a new risk measure was proposed by Chen and Wang in [15] in 2008. They proposed a coherent risk measure different from the previous ones, a measure that will take into consideration simultaneously, but in different ways, both negative and positive deviations from the expected return: a two-sided coherent risk measure. It is known that in the real world financial assets are not normally distributed, on the contrary they can present skewness and fat tails (leptokurtosis). This new risk measure can describe and control these characteristics and so reflect the attitude of the investor. The two-sides risk measure provides some advantages. First of all, it exploits the whole domain loss distribution information, which makes it superior for finding robust and stable investment decisions. Robust with respect to both trading sides and stable with respect to the estimation error. Second, it reflects the investor's risk attitude and controls the asymmetry and fat tails of the distribution, by suitably selecting the convex combination coefficient and the order of the norm of the downside random loss. Finally, it is easily applicable to find the optimal portfolio and it is easy to compute its value. We will analyze in details this measure of risk in the next chapters.

Chapter 2

Genetic Algorithms and Particle Swarm Optimization

This chapter will begin with a general description of metaheuristics. Then I will focus in particular in two bio-inspired metaheuristics: Genetic Algorithms and Particle Swarm Optimization.

2.1 Metaheuristics

Many optimization problems consists in searching the best configuration possible of a set of variables to achieve a goal. There exist a variety of exact methods to solve a large number of real-world problems like these, but they tend to be non-flexible in particular for complex and/or large problems. The main problem with portfolio optimization is dealing with huge amount of data available and limited computation time.

To overcome these problems, metaheuristics have been developed. The term metaheuristic derives form heuristic which means "to find or to discover by trial and error" [18]. With the use of heuristics it is possible to find, in a reasonable amount of time, a solution to complicated optimization problems. The solution found, though, has no guarantee to be the optimal one. Heuristics provide good solutions which are easily reachable. As Yang said in [18]:

"Heuristics is a way by trial and error to produce acceptable solutions to a complex problem in a reasonably practical time. The complexity of the problem of interest makes it impossible to search every possible solution or combination, the aim is to find good feasible solution in an acceptable timescale".

Metaheuristics are a development of heuristics. The prefix *meta* in Greek means "beyond", in the sense of higher level, and they generally perform better than simple

heuristics. In literature there is not a common definition for metaheuristics, I will use the one provided by Sörensen and Glover in [19]:

"A metaheuristic is a high-level problem-independent algorithmic framework that provides a set of guidelines or strategies to develop heuristic optimization algorithms".

Metaheuristics are good techniques to solve NP-hard problems, like, for instance, complex portfolio selection problems, because they are not subject to combinatorial explosion (while the time required to compute the optimal solution of a NP-hard problem increases exponentially with the size of the problem). Compared to exact methods, metaheuristics are more flexible since they are designed in general terms. Biologically inspired algorithms can be adapted to fit the needs of many real-life optimization problems. However, problem-specific adaptation is needed to achieve a good performance. All types of metaheuristics are strategies that "guide" the search process and their goal is to efficiently explore the search space to find (near)-optimal solutions. The idea is to have a practical and efficient algorithm that works most of the times.

Metaheuristics have two important components that provide a dynamic balance: diversification and intensification. Diversification refers to the generation of diverse solutions in order to explore the search space on the global scale. Intensification, on the other hand, means to focus on the search in the local region by exploiting the information that a current good solution is found in this region [18]. In other words, diversification increases the diversity of the solution and ensures that it will not be trapped in the local optima, while intensification ensures that the solution will converge to the optimality. Global optimality is reached with a good mix of these two components.

Sometimes the terms exploration and exploitation are used instead of diversification and intensification. Exploration and exploitation have a slightly different meaning, they refer to short-term strategies tied to randomness. Diversification and intensification, instead, are medium and long-term strategies that are based on the usage of memory.

There are many ways in which metaheuristics can be classified; one important way is related to the components of diversification and intensification. Metaheuristics can be classified in two categories: trajectory-based and population-based.

- **Trajectory-based**

"The search process performed by these methods is characterized by a trajectory in the search space" [21]. This means that, since the algorithm is able to move in the search space to look for better solutions, a successor solution may

belong to the neighborhood of the current solution or may not. The algorithm starts from an initial solution and trace a trajectory in the space. From the initial solution, at each step of the search the current solution is replaced by a better solution. This allows to converge towards the local optima. This basic search is called iterative improvement. Trajectory-based metaheuristics are intensification-oriented since they search for locally optimal solutions. Some metaheuristics that belong to this category are: Simulated Annealing (SA), Tabu Search (TS) and Variable Neighborhood Search (VNS).

- **Population-based**

Population-based methods do not take into account a single solution like trajectory-based methods; they deal with a set, called population, of solutions. In this way, they provide a natural, intrinsic way to explore the search space. This methods start with an initial population which is randomly generated. The algorithm uses an iterative process which, at each generation process, substitutes the whole or a part of the population with newly generated individuals that are generally better. Since the main ability of population-based metaheuristics is the diversification through the search space, they are diversification-oriented. In this category we find the metaheuristics that we are going to study later in this chapter: Genetic Algorithms (GA), that are part of Evolutionary Algorithms (EA) and Particle Swarm Optimization (PSO). Other important metaheuristics are Ant Colony Optimization (ACO) and Artificial Bee Colony (ABC).

2.1.1 History Overview

Before proceeding more in detail with GAs and PSO, I would like to make a brief overview of the history of metaheuristics. Humans have always used heuristics throughout history to solve problems: the human approach to solve optimization problems has always been heuristic or metaheuristic. Although metaheuristics' omnipresence in the world, the scientific study of (meta-)heuristics is a modern phenomenon and it is difficult to exactly know when the first metaheuristic method was used. Probably the first person to use it was Alan Turing⁵ during World War II. Turing used heuristic algorithms to decode the German Enigma ciphers at Bletchley Park. He was one of the firsts to develop the ideas of machine intelligence, neural networks and evolutionary algorithms.

⁵Alan Turing was an English computer scientist, mathematician, logician, cryptanalyst, philosopher and theoretical biologist. He is considered the father of theoretical computer science and artificial intelligence.

An important evolution in the study of metaheuristics happened in 1960s and 1970s. In these decades, there has been the development of evolutionary algorithms. In the early 60s, crossover and recombination manipulation were used for the first time to study the adapted system by Holland. He developed GAs and published his book in 1975. In the same year also De Jong published his dissertation in which he shows the power and potential of genetic algorithms for a wide range of objective functions.

The most exciting time for the study of metaheuristics have been the 80s and 90s. In this period, an optimization technique inspired by the annealing process of metals was developed. This technique is now known as SA, and its inventors were S. Kirkpatrick, C. D. Gellat and M. P. Vecchi.

Only later in 1995 PSO was developed by James Kennedy and Russel C. Eberhart. PSO is a technique to solve optimization problems that is inspired by swarm intelligence of fish and birds. The studies of metaheuristics are far from concluded, new studies are constantly developed and old ones updated.

2.2 Genetic Algorithms

In this section I am going to analyse GAs in details. As said before, GAs were developed by Holland in the 60s. In 1975 Holland published his book *Adaptation in Natural and Artificial Systems*, in which he presented the genetic algorithm as an abstraction of biological evolution and gave a theoretical framework for adaptation under the GAs. The idea of Holland was to study the phenomenon of adaptation as it occurs in nature and to develop ways in which the mechanisms of natural adaptation might be imported into computer systems. His original purpose was not to create algorithms to solve specific problems.

In order to understand how GAs work, it is useful to have a little background on heredity at the cellular level. The basic unit of heredity is the *gene*. Genes are contained in the *chromosomes* which form the DNA. The DNA is a system of base-pair sequences that determine the sequence of enzymes and other proteins in an organism. This sequence remain unchanged in each organism and it is its *genetic code*. Roughly speaking, each gene is encoding a *trait*, e.g. the eye colour, and all possible settings of that trait are called *alleles*, e.g. blue, brown, green. In human beings, chromosomes are arrayed in pairs; this is called *diploid*. During the sexual reproduction of diploid organisms, *recombination* or *crossover* occurs: in each parent, genes are exchanged between each pair of chromosomes in order to form a single chromosome, the *gamete*, and then gametes from the two parents

pair up to create a full set of diploid chromosomes. The offspring is then subject to mutation. With mutation, each single nucleotide (elementary bits of DNA) is changed from parents to offspring. The fitness of an organism is usually defined as the probability that the organism will live to reproduce or as a function of the number of offspring the organism has [24].

GAs are part of evolutionary methodologies, that are biologically inspired algorithms drawn from an evolutionary metaphor. These evolutionary methodologies are inspired by biological evolution. As said in [17]:

"In biological evolution species are positively or negatively selected depending on their relative success in surviving and reproducing in their current environment. Differential survival and variety of generation during reproduction provide the engine for evolution."

GA is an optimization and search technique based on the principles of genetics and natural selection, with global search potential. GAs apply a pseudo-Darwinian process to evolve good solutions to real-world problems. A GA allows a population composed of many individuals to evolve under specified selection rules to a state that maximizes the "fitness", i.e. minimizes a prefixed cost function [23]. In short, the methodology evolved by Holland begins with a population of chromosomes that, with the use of specific GA's operators, turn into a new population, a new generation. GAs are composed by strings (chromosomes) that are in turn composed by bit (genes). The values that the bit can take are called alleles and are 0 or 1, this because the numerical system used to define the bit is the binary one. There are specific basic operators that apply to GAs, that are selection, crossover and mutation:

- **Selection:** chromosomes for reproduction are selected in the population. The chromosomes that fits best are more likely to be selected more times to reproduce;
- **Crossover:** a locus (position) is choosen randomly by this operator and the subsequences before and after that locus are exchanged between two chromosomes to create two offspring;
- **Mutation:** Some of the bits in a chromosome are randomly turned. Mutation occurs with a probability that generally is very low (e.g. 0.001) and each bit position can be mutated. For example consider the string 10000110, it might be mutated in its third position to yield 10100110.

The chromosomes are strings composed by sequences of bit (genes) that can be interpreted as potential solutions of a specific problem. Each potential solution is

encoded as a binary string and the quality of each string is determined by a fitness function, which maps each string to a number representing its quality or fitness. The fitness function is problem specific and it values the most adapt solutions (chromosomes) to solve the problem.

Like biological genotypes, that encodes the results of past evolutionary trials, GAs encode a history, or memory, of the current population when future population are developed.

Before going further in details of how GAs work, here are listed some of their advantages [23]:

- Optimize both continuous and discrete variables;
- Doesn't require derivative information;
- Simultaneously searches from a wide sampling of the cost surface;
- Deals with a large number of variables;
- Optimize variables with extremely complex cost surface;
- They can jump out of a local minimum;
- Provide a list of near optimum solutions, not just a single one;
- Since the bits use the binary system, GAs are able to encode the variables and perform the optimization on the encoded variables;
- Works with numerically generated data, experimental data or analytical functions.

Evolutionary algorithms, which includes GA, can be formalized as follows, as done by Brabazon and O'Neill in [17]:

$$x[t + 1] = r(v(s(x[t])))$$

where $x[t]$ is the population of encodings at iteration t , $v(\cdot)$ is the random variation operator (crossover and mutation), $s(\cdot)$ is the selection for mating operator and $r(\cdot)$ is the replacement selection operator. After the initial population of strings encoding solutions has been obtained and evaluated, a reproductive process is applied in which the encodings corresponding to the better-quality solutions have a higher chance of being selected for propagation of their genes into the next generation.

2.2.1 A simple GA

In this section I am going to explain how a simple GA works. As said in [17]:

"The canonical GA can be described as an algorithm that turns one population of candidate encodings and corresponding solutions into another using a number of stochastic operators. Selection exploits information in the current population, concentrating interest on high-fitness solutions. Crossover and mutation perturb these solutions in an attempt to uncover better solutions. Mutation does this by introducing new gene values into the population, while crossover allows the recombination of fragments of existing solutions to create new ones. It is important to notice that the evolutionary process of a GA operates on the encodings of the solutions, rather than directly on the solutions themselves."

Given a defined problem to solve, the GA works as follows:

- i. *Initialization*: the initial population, of n encodings corresponding to n candidate solutions, is randomly constructed. It is also determined how the solution is to be encoded as a string and the fitness function is specified;
- ii. *Valuation*: each time a population has been initialised or each time a new generation has been created, the fitness function provides a valuation on each single potential solution;
- iii. *Selection*: the process of selection defines which pair of encodings corresponding to candidate solutions (the *parents*), from the existing population, have to be used to create a new generation of solutions. Potential solutions with higher fitness function have more probability to be chosen;
- iv. *Recombination*: a crossover process on the encodings on the selected parent solutions is performed with probability p_{cross} , to produce two new child solutions. Recombination is made to obtain possible solutions with higher levels of fitness. Parents and children will have different chromosomes, the new generation will have the genes of the chromosomes of the parents recombined with each other;
- v. *Mutation*: a mutation process is applied to each element of the encodings of the two child solutions with probability p_{mut} . While recombination exploits characteristics already present in the parents' chromosomes, mutation alters in a random way the genes. That is, with mutation a kind of random walk is applied to the potential solution that is subject to it, in order to "explore" the solution space;

- vi. *Replacement*: the new generation of potential solutions substitute, partly or completely, the initial population;
- vii. Repeat from step ii to step vii until the desired population fitness level is reached or until a predetermined number of generations have elapsed.

2.2.2 How do GAs work

In this section I am going to describe in more details some of the selection, crossover and mutation operators that are commonly used in GAs.

Selection Methods

When designing a GA for a specific problem, a key issue is to manage the trade-off between exploration and exploitation. The choice of the selection process and of the mutation and recombination operators determine this trade-off. If we take into consideration the classification of selection methods proposed by K. Sastry, G. Kendall and D. Goldberg in [25], the selection procedures can be divided into two families: Fitness Proportionate Selection and Ordinal Selection.

- *Fitness Proportionate Selection*

The simplest and original selection method for reproduction is the the fitness-proportionate selection. In this method, the selection process is directed towards good members of the current population. This is because the probability that a specific member is selected is directly related to its relative fitness. This family includes methods such as stochastic universal selection and roulette-wheel selection. In order to explain how roulette-wheel selection works, lets imagine a biased roulette wheel, in which good solutions have a larger slot size than less fit solutions; in particular, each individual in the population has a slot with the size proportional to its fitness. To know exactly which proportion of the wheel has to be assigned to a particular chromosome, it is necessary to compute the relative weight of its fitness function. In order to do it we have to evaluate the fitness f_i of each individual in the population and then compute the probability p_i of selecting each member of the population :

$$p_i = f_i / \sum_{j=1}^n f_j$$

where n is the population size. To calculate the cumulative probability q_i for

each individual the formula is the following:

$$q_i = \sum_{j=1}^n p_j.$$

Everytime the wheel spins, a chromosome is extracted and is going to create the mating pool. To spin the wheel means to extract a uniform random number $r \in [0, 1]$. If $r < q_1$ the first individual x_1 is selected, otherwise an individual x_i such that $q_{i-1} < r \leq q_i$ is selected. The wheel is spin more times until $2n$ parents are selected and n candidates are created for the mating pool.

The fitness proportionate selection methods though can force too much the selection of individuals with high-values fitness functions in the current population. Usually, there is a high variance in the fitness of solutions, especially in the first stages, where high fitness solutions are a small number. This family of processes cause the loss of diversity in the population too soon, they tend to reduce the exploration of the search space.

- *Ordinal Selection*

This family of methods have been designed to overcome the problems of fitness-proportionate selection. Two major methods included in this category are tournament selection and truncation selection. Under tournament selection, s individuals are choosen randomly, with or without replacement, and entered into a tournament against each other. The fittest individual in the group of k individuals wins the tournament and is selected to act as a parent. The most used value of s is 2, lower values of s provide lower selection pressure, while higher values provide higher selection pressure. Using this selection method, to choose n individuals are required n tournaments. Truncation selection, instead, works by sorting individuals belonging to the same population in a ranking, where the first individual is the one with the highest fitness value. A certain position p , with $p = 1/2, 1/3, \dots$, of the individuals with higher fitness value is selected and reproduced $1/p$ times in the mating pool.

Crossover Operators

After the chromosomes that are going to form the new generation are choosen, the crossover or recombination operator is applied to them. Crossover serves two purposes in the GA: it allows the inheritance of good genes or partial solution encodings by the offspring of parents and it serves to reduce the search space to

regions of greater promise. Crossover stops to generate novelty once all members of the population converges to a single string form. There is a large number of ways in which crossover can be implemented. A common feature in most of these ways is that, the two "parent" individuals are randomly selected and are recombined with a probability p_c (the crossover probability). That is, a uniform random number u is generated and confronted with the probability p_c . Then if $u \leq p_c$, the two randomly selected individuals undergo recombination. Otherwise, if $u > p_c$, the two individuals are not subject to recombination and the offspring will be copies of the parents. The value of the probability p_c is generally setted experimentally, during the setting phase of the GA. Some of the most important ways to implement crossover are listed following.

- **k -point Crossover:** Firstly, two parents are selected randomly from the mating pool. In one-point crossover, a crossover point is selected at random over the string length, and the alleles on one side of the site are exchanged between the individuals [25]. When $k = 2$, the crossover sites randomly selected are two. In this case the alleles between two sites are exchanged between the parents. This method can be extended to k crossover sites. An example of one-point and two-point crossover is shown in figure 2.1.

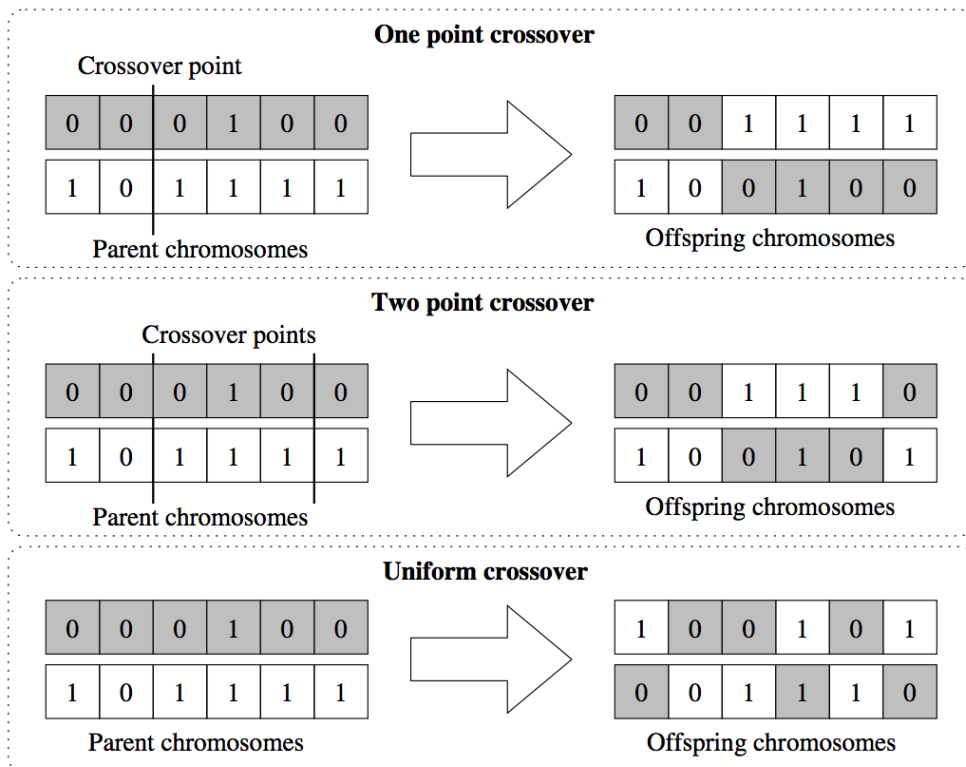


Figure 2.1: One-point, two-point, and uniform crossover methods.

- **Uniform Crossover:** In this popular form of crossover, the two parent strings are compared element by element. In particular, each gene belonging to the first parent chromosome is exchanged with the corresponding gene belonging to the second parent chromosome with a certain probability p_e , which is known as the swapping probability. Usually p_e is set equal to 0.5. An example of this crossover method can also be seen in figure 2.1.
- **Uniform Order-Based Crossover:** Let's consider two parents, P_1 and P_2 , that are randomly chosen, and a string of 0 and 1 randomly generated, of the same length of the strings of the parents P_1 and P_2 . The process for the creation of the first offspring C_1 is the following: firstly look at parent P_1 and the string of 0 and 1. Where the string has value 1, the offspring C_1 is going to take the same gene of parent P_1 , in the same position. Now C_1 is partially completed, with some spots that are temporarily empty. In order to complete C_1 , it is necessary to look which are the genes of P_1 corresponding to the value 0 in the string. These genes have to be included in the offspring C_1 in the order as they appear in parent P_2 . Now that there are no spots empty, offspring C_1 is created. Offspring C_2 is created following the same procedure, but taking as referent parent P_2 . Figure 2.2 exemplify this crossover method.

Parent P_1	A	B	C	D	E	F	G
Parent P_2	E	B	D	C	F	G	A
Template	0	1	1	0	0	1	0
Child C_1	E	B	C	D	G	F	A
Child C_2	A	B	D	C	E	G	F

Figure 2.2: Uniform Order-Based Crossover method.

- **Order-Based Crossover:** This method is a variation of the uniform order-based crossover. As always, it starts selecting at random two parents from the mating pool. In this method it is not considered a string of 0 and 1, but two crossover sites, like in the k -point crossover. The two crossover sites are randomly generated in the parents P_1 and P_2 . Let's consider the creation of offspring C_1 . Firstly, look at the genes in P_1 positioned in between the two crossover sites and copy them in C_1 in the same position. Then the empty spots are going to be filled starting from the second crossover site. Now look

at the alternative parent P_2 and, in the order they appear, copy the genes that are not yet present in offspring C_1 , in order to fill all the empty spots. As before, offspring C_2 is created in the same way. A graphical exemplification is shown in figure 2.3.

Parent P_1	A	B	C	D	E	F	G
Parent P_2	C	B	G	E	F	D	A

Child C_1	?	?	C	D	E	?	?
Child C_2	?	?	G	E	F	?	?

Child C_1	F	A	C	D	E	B	G
Child C_2	C	D	G	E	F	A	B

Figure 2.3: Order-Based Crossover method.

- **Partially Matched Crossover (PMX):** In this method, as in order-based crossover, two parents P_1 and P_2 and two crossover sites are randomly generated. To create the two offspring C_1 and C_2 one has to start from looking at the genes in between the two crossover sites of both parents, P_1 and P_2 . Now, look at the genes in between the crossover sites of P_1 and pair them with the genes in the same position of parent P_2 , in this way one couple for each gene in between the crossover sites is created. Observing, for example, Figure 2.4, three couples are created: 5 and 2, 6 and 3 and 7 and 10. In order to create the offspring C_1 and C_2 , it is necessary to swap the values of the couples just created. Doing so, some spots will remain empty. To fill them just copy the gene present in the same position of the respective parent: P_1 for C_1 and P_2 for C_2 . To continue with the example, for offspring C_1 , just swap 5 with 2, 3 with 6 and 10 with 7, and for the remaining empty spots copy the values in the same position of P_1 .
- **Cycle Crossover:** We again start with the random selection of two parents. Let's consider the creation of offspring C_1 . To start, look at the gene in the first position of parent P_1 and copy it in the same position in offspring C_1 . Then, look at the gene present in the same position (the first one) of parent P_2 . Now, copy in offspring C_1 the gene present in the first position of P_2 , but

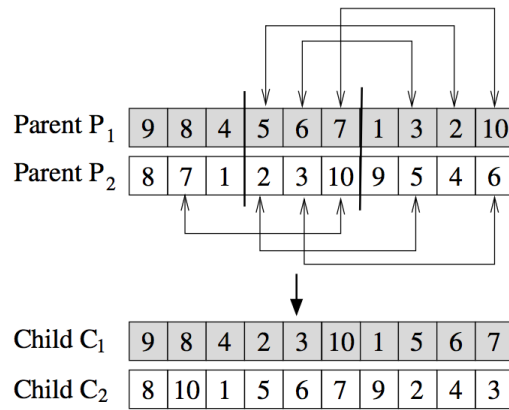


Figure 2.4: Partially Matched Crossover method.

in the position in which appears in parent P_1 . Then continue to look at the gene present in parent P_2 in this position and copy it in C_1 in the position in which it appears in P_1 . This procedure is repeated cyclically until we enter in a loop going back to the first gene considered. The spaces that are left empty in offspring C_1 , are filled up by copying the genes in the same position of parent P_2 . Offspring C_2 is generated in the same way, just exchanging the roles between parent P_1 and P_2 . An example of this method is shown in Figure 2.5.

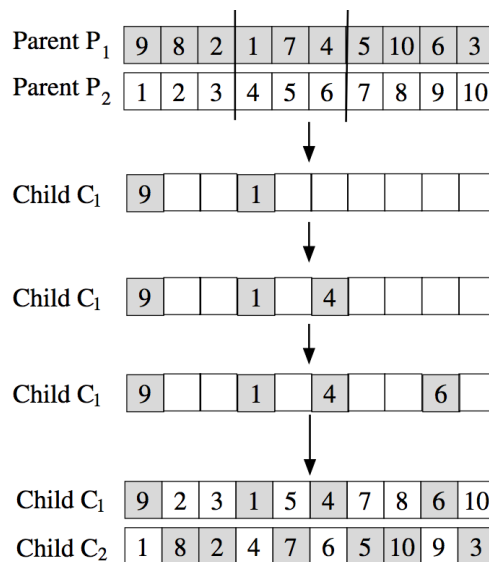


Figure 2.5: Cycle Crossover method.

Mutation Operators

In this section, we are going to analyze mutation, the secondary operator in GAs. Crossover operators, described above, allow only to shuffle the genes already present in the potential solutions and in the mating pool. Think for example at one-point crossover. If two or more chromosomes have the same allele in the same position, the next generations will also have the same allele in the same position. This consideration shows one limit of the crossover operator, it can be unable to alter the structure of a population of chromosomes, preventing so a wide search in the space of possible solutions.

Mutation is used to add diversity to the population and to ensure that it is possible to explore the entire search space. In other words, it introduces traits not present in the original population and keeps the GA from converging too fast before sampling the entire search space. With mutation operators, the value of a gene is altered: this means that the allele corresponding to that gene is modified. This operator is applied with a probability, p_m , that is called the mutation probability. One of the most common mutations is the bit-flip mutation, in which each bit in a binary string is changed, 0 becomes 1 and vice versa, with probability p_m . More articulated variants can be developed to meet problem-specific needs, and are often used in literature.

Replacement

Now that, using crossover and mutation, new offspring solutions are created, we need to introduce them into the population of the parents. We expect and hope that the population will gradually increase its fitness. The reason we expect that is because the parent chromosomes have already been selected according to their fitness, so the offspring, which includes the parents which did not undergo crossover and mutation, are among the fittest in the population. There are many ways in which the new generation of chromosomes can replace the one of parent chromosomes. The goal is to have a population with higher level of fitness. There are three main replacement ways and are listed following:

- **Delete-all:** This is the most common and easily implementable technique. The previous population, the one from which the parents chromosomes are selected, is completely erased and substituted by the children chromosomes. The new population is constituted just by the offspring. One reason why it is easy to implement is that this technique does not require any parameter to define the proportion of parents chromosomes and children chromosomes present in the new population.

- **Steady-state:** With this technique, the new population of potential solutions is composed by both individuals belonging to the population from which the parents individuals were selected, and from the children individuals. $n_1 < n$ old members are deleted and replaced with n_1 new members. To define the number to delete and replace, n_1 , a parameter is defined. This technique needs also another parameter that defines a criterion with which to choose the individuals to delete from the current population (the worst individuals can be deleted, can be picked at random or can be deleted the individuals that were used as parents).
- **Steady-state no-duplicated:** This technique, as the name suggests, is a variant of the steady-state technique. The operating mechanism is the same, but this time the algorithm checks that no duplicate chromosomes are added to the population. With this technique, the computational overhead is increased, but a wider exploration of the search space is ensured.

2.2.3 The Schema Theorem

So far I have described how GAs and their operators work. In this section I am going to provide some theoretical support on the efficiency of GAs, in order to explain their widespread use. I am going to start with the original theory of GAs formulated by Holland that assumes that [24]:

"At every general level of description, GAs work by discovering, emphasizing, and recombining good "building blocks" of solutions in a highly parallel fashion. The idea here is that good solutions tend to be made up of good building blocks - combination of bit values that confer higher fitness on the strings in which they are present."

Building blocks have a very important role in the workings of GAs. To formalize the notion of "building blocks", Holland introduced another notion: *schemas*. Schemas are implicitly the building blocks that the GA processes effectively under the operators of selection, mutation, and crossover. A schema can be represented as a bit strings of a potential solution that can be described as a template made up of zeroes, ones and asterisks (0, 1, *). The asterisks are wild cards that can assume both values, 0 and 1. So, with a single schema it is possible to give a common representation to a high number of potential solutions, to all the solutions that contain the schema itself. From these concepts the *Schema Theorem* is derived. Holland presented the theorem in his work in 1975 and it is the main theoretical foundation supporting the use of GAs. Through this theorem, the (sub-)optimization properties of GAs are explained, proving that, under defined

constraint, potential solutions with higher fitness value tend to increase exponentially within the population. In particular, Holland, given a generation at iteration t , provides a formula for the calculation of the minimum number of schemas of a certain type that will be found in the next generation at iteration $(t + 1)$. To better understand, let's see an example of schema. I will use the notation used by Mitchell in [24], each schema is denoted as H that stands for "hyperplane", because schemas define planes of various dimensions. Let's consider, for example, the schema $H_i = 1 * * 1$, it represents the following potential solutions:

- 1001
- 1011
- 1101
- 1111

Schemas can be distinguished on the basis of two main characteristics: the order and the defining length. The order is the number of alleles defined (non-asterisks) in the schema, in the example, H_i is of order two because it has two defined bits. The defining length is the distance between its outermost defined bits, for H_i the distance between the first and last defined allele is 3.

Before continuing with the description of the theorem, it is useful to introduce the notion of implicit parallelism. This concept indicates the ability of GAs to analyze simultaneously, i.e. parallelly, different schemas concerning single solution which constitute the population. It is explained in [24] as follows:

"At a given generation, while the GA is explicitly evaluating the fitness of n strings in the population, it is actually implicitly estimating the average fitness of a much larger number of schemas, where the average fitness of a schema is defined to be the average fitness of all possible instances of a schema."

To better explain this definition, consider a general solution "000". All the schemas that can be associated to this solution are:

- * * *
- * * 0
- * 0 *
- 0 * *
- 0 0 *

- *00
- 0*0
- 000

The schemas one can obtain from a solution of length 3 are $2^3 = 8$. In general, a solution of length l can be represented by 2^l schemas. Thus, any given population of n strings contains instances between 2^l and $n \times 2^l$ different schemas. The number of schemas that the AG has to process within a population of n potential solutions will be the minimum (2^l) when the n chromosomes are all identical, and the maximum ($n \times 2^l$) when the n solutions are all different among themselves. It is important to note that schemas are not explicitly represented or evaluated by the GA and that the estimates of schemas' average fitness are not calculated or stored by the GA. However, in terms of the increase and decrease in numbers of instances of given schemas in the population, the GA's behavior can be described as it actually was calculating and storing the estimates of schemas' average fitness.

The approximate dynamics of the increase and decrease in the schema instances can be calculated as follows [24].

Let H be a schema with at least one instance present in the population at time t . Let $m(H, t)$ be the number of instances of H at time t , and let $\hat{U}(H, t)$ be the observed average fitness of instances of H in the population at time t . We want to calculate the expected number of instances of H at time $t + 1$, denoted as $E(m(H, t + 1))$. Assume that fitness proportionate selection is carried out, meaning that the expected number of offspring of a string x is equal to $p_i = f_i / \sum_{j=1}^n f_j$ (as described in the previous section). This formula can be rewritten in the following way:

$$f(x)/\bar{f}(t)$$

where $f(x)$ is the fitness of x and $\bar{f}(t)$ is the average fitness of the population at time t . Then, assuming x is in the population at time t and (for now) ignoring the effects of crossover and mutation, we have by definition:

$$E(m(H, t + 1)) = \sum_{x \in H} f(x)/\bar{f}(t) = (\hat{u}(H, t)/\bar{f}(t))m(H, t). \quad (2.1)$$

Even though the GA does not calculate $\hat{U}(H, t)$ explicitly, the increases or decreases of schema instances in the population depend on this quantity. Now let's take into consideration the effects of crossover and mutation. These two operators can both create and destroy instances of H . In this work I will consider just the destructive effects, the ones that decrease the number of instances of H at time $t + 1$. The

inclusion of the destructive effect will modify just the right part of the formula (2.1).

Let p_c be the probability that single-point crossover will be applied to a string, and suppose that an instance of schema H is picked to be a parent. Then, if one of the offspring is also an instance of schema H , H is said to survive under single-point crossover. To make sure that H will survive the single-point crossover, a lower bound, $S_c(H)$, is given on the probability:

$$S_c(H) \geq 1 - p_c \left(\frac{d(H)}{l-1} \right)$$

where l is the length of bit strings in the search space and $d(H)$ is the defining length. The fraction of the string that H occupies is multiplied by the crossover probability to obtain an upper bound on the probability that it will be destroyed. Subtracting this value from one gives a lower bound on the probability of survival $S_c(H)$. The probability to survive crossover is higher for shorter schemas.

Regarding mutation, it is possible to calculate its destructive effect as follows. Let p_m be the probability of any bit being mutated, and let $S_m(H)$ be the probability that the schema H will survive under mutation of an instance of H . It is possible to prove that:

$$S_m(H) = (1 - p_m)^{o(H)}$$

where $o(H)$ is the order of H . The probability that a bit of schema H will be mutated is the quantity $1 - p_m$ multiplied by itself $o(H)$ times. The probability to survive mutation is higher for lower-order schemas.

Joining the disruptive effects of crossover and mutation operators to equation (2.1), leads to the *Schema Theorem*:

$$E(m(H, t+1)) \geq \frac{\hat{u}(H, t)}{\bar{f}(t)} m(H, t) \left(1 - p_c \frac{d(H)}{l-1} \right) [(1 - p_m)^{o(H)}].$$

This theorem describes the expected growth of a schema from one generation to the next one. As introduced before, this theorem implies that short, low-order schemas whose average fitness remains above the mean will receive exponentially increasing numbers of samples over time, since the number of samples of those schemas that are not disrupted and remain above average in fitness increases by a factor of $\hat{U}(H, t)/f(t)$ at each generation.

Despite its destructive effect, crossover is considered a major source of the power of GAs, because it has the ability to recombine instances of good schemas to form instances of equally good or better higher-order schemas. This is also known as the

2.3 Particle Swarm Optimization

Particle Swarm Optimization (PSO) is a population-based optimization technique that was developed by J. Kennedy and R. Eberhart in 1995. In [28] they introduced a concept for the optimization of continuous nonlinear functions using particle swarm methodology. This specific metaheuristic is drawn from a swarm metaphor. There exists two popular variants of swarm models, those that are inspired by the behaviour of social insects (i.e. ant colonies), and those that are inspired by the sociological behaviour of a group of people or by the flocking behaviour of birds. PSO belongs to the second category. As stated by Kennedy and Eberhart, PSO has roots in two main component methodologies: artificial life in general and birds flocking, fish schooling, and swarming theory in particular. It is also tied to other methodologies, like Genetic Algorithms and Evolutionary Programming. To be more precise, the cornerstones of the PSO are the social concepts and the Swarm Intelligence (SI) principles. The social concepts can be expressed as: "human intelligence results from social interaction" and "culture and cognition are inseparable consequence of human society". The first concept refers to the fact that individuals learn from experience, they imitate each other, they adapt to the environment and suchlike. The second concept is defined in [30] like this: "culture is generated when individuals become more similar due to mutual social learning. The sweep of culture allows individuals to move towards more adaptive patterns of behaviour". Before continuing it is necessary to dedicate some time to the concept of SI, this is done in the following section.

2.3.1 Swarm Intelligence

The term swarm indicates an aggregation of animals performing in collective behaviour; a large number of homogeneous, unsophisticated agents that interact locally among themselves and their environment, without any central control to yield a global behaviour to emerge. Generally, the single agents are not smart individually but the swarm is smart.

The first definition of SI was given by Beni, Hackwood and Wang: "many simple agents occupy one- or two-dimensional environments to generate patterns and self organize through nearest-neighbor interactions". From this, SI can be defined as a collective behaviour of decentralized and self-organized swarms.

The most interesting thing is how the agents connect their individual behaviour

with a collective performance. Many aspects underlying cooperation are self-organised. As said in [34] "theories of self-organization, originally developed in the context of physics and chemistry to describe the emergence of macroscopic patterns out of processes and interactions defined at the microscopic level, can be extended to social insects to show that complex collective behaviour may emerge from interactions among individuals that exhibit simple behaviour: in these cases, there is no need to invoke individual complexity to explain complex collective behaviour". It is possible to say that a social insect colony is a decentralized problem-solving system composed by many relatively simple agents that interact among them. An important feature of social insects is that they are able to solve a problem in a flexible and robust way. Flexibility refers to the fact that they can adapt to changing environments, while robustness refers to the ability of the swarm to function even though some agents fail to perform their tasks. Self-organization consists in a set of dynamical mechanisms whereby structures appear at the global level of a system from interactions among its lower-level components [34]. The four basis of self-organization proposed by Bonabeau are:

- *Positive Feedback*: consists in empirical rules that promote the creation of convenient structures. This principle includes reinforcement and recruitment⁶;
- *Negative Feedback*: it serves to counterbalance positive feedback. The swarm can reach saturation, exhaustion or competition⁷, negative feedback helps in stabilizing the collective pattern;
- *Fluctuations*: one of the basis self-organization relies on, is the amplification of fluctuations. Fluctuations are represented by random walks, errors and so on, and they are useful for the swarm. They enables the discovery of new solutions, and can be the start of new structures that nucleate and grow. Furthermore, structures will not be penalized by randomness, they will emerge despite it;
- *Multiple Interactions*: Self-organization require the swarm to be composed by multiple agents that tolerate each other. Moreover, the agents have to be able to make use of their personal results as well as others' results.

According to Valle et al. in [30], to complete the definition of SI, and for a swarm

⁶Reinforcement and recruitment in nature refers to the find of a food source. When a food source is found, the individual reports a positive feedback to the swarm in order to recrute more individuals and reinforce food supply.

⁷In the case of food supply, saturation, exahustion and competition refers to satiation, food source exhaustion, crowding at the food source or competition between food sources.

to be intelligent, five fundamental principles have to be taken into consideration in case of an optimization problem:

- *Proximity Principle*: simple time and space computations have to be carried out by the population;
- *Quality Principle*: the agents should be able to respond to different factors present in the search space;
- *Diverse Responce Principle*: the agents, in doing their activities, do not have to take ways that are excessively narrow;
- *Stability Principle*: when the environment change, the swarm has to keep its behaviour stable, it does not have to change it when changes in the environment occur;
- *Adaptability Principle*: the behaviour mode has to be changed when it is worth the computational price.

Now that the concept of Swarm Intelligence is clear, I will continue with the description of particle swarm optimization and how it works.

In PSO there is a number of autonomous simple entities, called the *particles*, that are randomly generated and placed in the search space of an optimization problem of a function. Each particle is a potential solution to the problem and evaluates the objective function at its current location. This means that each particle is associated to a velocity and to a location in the search space. All the potential solutions are contained in the swarm of particles that move in the search space on the attempt to uncover better solutions. So, a swarm is composed by N particles that move (fly) around a D -dimensional search space. Where N is the number of possible solutions of a given problem and D is the number of variables of that given problem.

A particle determines its movements through the search space by the memory of the current and best location that it has found so far (*pbest*), and the best location found to date by all the particles in the population (*gbest*). The quality of the position of a particle is measured by the fitness function. After all particles have moved, the next iteration takes place, and eventually the whole swarm will move closer to an optimum of the fitness function. A fundamental concept of PSO is the experience-sharing behavior. The experience of a single particle is constantly communicated to the rest of the swarm, so the swarm can move to the most promising areas and the best global position detected so far.

2.3.2 How does a PSO algorithm works

In this section, I will present how does the algorithm for PSO works, as done in [17]. The algorithm described is the one for the canonical continuous version of the PSO. The steps to follow are:

- i. Initialise each particle in the population. This is made selecting at random values for its location and velocity vectors;
- ii. Calculate the fitness value for each particle;
- iii. Set the best location of each single particle (*pbest*), and the best location of the whole swarm (*gbest*);
- iv. Calculate the new velocity for each particle;
- v. Update the position of each particle;
- vi. Calculate the new fitness value of each particle. If the new fitness value is higher than the previous one, modify the location of *pbest*;
- vii. When all particles have been updated, look for the particle with the highest value of fitness, and if it is higher than the previous *gbest*, modify it.
- viii. Repeat from step iv to step vii until a stopping criterion is met.

Now, I am going to describe more in details the mechanisms needed to compute the steps. A. Brabazon and M. O'Neill in [17] describe this algorithm as follows:

"At the start of the algorithm, the *pbest* for each particle is set at its initial location, and *gbest* is set to the location of the best of the *pbest*. In each iteration of the algorithm, a particle is stochastically accelerated towards its previous best position and towards a neighborhood (global) best position, thereby forcing particles to continually search in the most-promising regions found so far in the solution space."

In the initialization phase, to each particle i is associated the current position in the search space $x_i \in \mathbb{R}^D$, the current velocity $v_i \in \mathbb{R}^D$, and the personal best position (*pbest*) in the search space $p_i \in \mathbb{R}^D$. The current position can be seen as a vector of coordinates representing a point in the space. This current position is calculated at each iteration of the algorithm and, if the new position is better than any that has been found so far, the coordinates are stored in the vector p_i . Location, velocity and position of each particle are updated during each iteration of the algorithm. Velocity and position of each particle are updated by the following equations:

$$\begin{cases} v_i^{k+1} = v_i^k + U(0, \phi_1) \otimes (p_i^k - x_i^k) + U(0, \phi_2) \otimes (p_g - x_i^k) \\ x_i^{k+1} = x_i^k + v_i^{k+1} \end{cases} \quad (2.2)$$

where

- x_i^k and v_i^k correspond to the current position and velocity of the i -th particle at step k ;
- p_g corresponds to $gbest$, the global best position of the swarm;
- $U(0, \phi_1)$ and $U(0, \phi_2)$ are two vectors of random numbers uniformly distributed in $[0, \phi_1]$ and $[0, \phi_2]$, which are randomly generated at each iteration, for each particle;
- \otimes is a component-wise multiplication.

The first equation is composed by three terms. The first one is the current velocity, v_i^k , it is also called "inertia". It serves to keep the particle moving in the direction it was originally heading. The second term is the cognitive component, represented by $U(0, \phi_1) \otimes (p_i^k - x_i^k)$. It is the memory of the particle, that leads the particle to return to the region of the search space where its individual fitness value is higher. The third term is called the social component: $U(0, \phi_2) \otimes (p_g - x_i^k)$. It takes into consideration the whole swarm, and makes each single particle move to the best region the swarm has found to date. The cognitive and social component are described in [29] as "attractive forces produced by springs of random stiffness, and we can approximately interpret the motion of the particles as the integration of Newton's second law⁸".

Once velocity and position are updated, also $pbest$ is updated, if necessary. Let $pbest_i = f(p_i)$ be the value of the fitness function evaluated in the personal best position visited by the i -th particle, and $gbest = f(p_g)$ be the fitness value of the global best position, p_g . The aim is to find new positions that improve the value of the fitness function. In the case of a maximization problem, $pbest$ is updated through the following equations:

$$p_i^{k+1} = \begin{cases} p_i^k & \text{if } f(x_i^{k+1}) < f(p_i^k) \\ x_i^{k+1} & \text{if } f(x_i^{k+1}) \geq f(p_i^k) \end{cases} .$$

In the case of a minimization problem the equation is vice versa and the value of the function has to be as low as possible.

⁸In an inertial reference frame, the vector sum of the forces \mathbf{F} on an object is equal to the mass m of that object multiplied by the acceleration \mathbf{a} of the object: $\mathbf{F}=m\mathbf{a}$ [36].

2.3.3 Parameters of the Algorithm

This basic version of the PSO algorithm needs a small number of parameters to be fixed. It is important to make several considerations, specially on the acceleration parameters and the maximum velocity, in order to prevent the explosion of the swarm and facilitate its convergence. The term explosion, in this case, refers to the fact that particles' velocities and positions can tend to infinity, creating the swarm to dissolve, explode. The first parameter needed is the size of the population, the number of particles in the swarm. This value is often set empirically, depending on the dimensionality and perceived difficulty of the problem.

Acceleration parameters

In equation (2.2), the terms ϕ_1 and ϕ_2 corresponds to the acceleration parameters. They "determine the magnitude of the random forces in the direction of the personal best and neighborhood best" [29]. In particular, ϕ_1 controls that each single particle moves towards its personal best p_i^k , and ϕ_2 controls that each particle moves toward the global best position p_g . The values of ϕ_1 and ϕ_2 have to be chosen carefully since the behaviour of the PSO can change radically with different values. Small values tend to limit the movement of the particles. The pattern of the trajectory of a particle will be similar to a sinusoidal waveform. Large values, on the other hand, will cause the particles to diverge. This leads complex paths of interwoven cyclic trajectories to appear and an increase of the frequency of the oscillations around the optimal point. Setting $\phi = \phi_1 + \phi_2$, in general the value of the variable ϕ has to be equal to 4. From this, a proposed good starting point is $\phi_1 = \phi_2 = 2$. Note that it is not necessary that ϕ_1 and ϕ_2 have equal value, since the weights for individual and group experience can vary according to the characteristics of the problem [30]. In the cases when $\phi > 4$, the trajectories tend to infinity. The acceleration parameter ϕ controls the strength of the term $(p - x)$, this means that a small value will have a weak effect, leading the particles to follow a wide path and they will be pulled back only after a large number of iterations. In the case of too high values, the steps will be limited by V_{max} , the maximum velocity.

Maximum velocity

The velocity at which each particle moves in the search space is adjusted by the algorithm at each iteration step. The velocity is a stochastic variable that is subject to creating an uncontrolled trajectory, that makes the particle follow wider cycles in the hyperplane. In the original version of PSO, each component of the velocity vector is restricted within the range $[-V_{max}, +V_{max}]$. In the majority of

the problems, the value of V_{max} is set empirically according to the characteristics of the specific problem. The value chosen for this range is very important and requires some care since it can have effects on the efficiency of the algorithm and it can influence the balance between exploration and exploitation. Small values can lead to an insufficient exploration of the search space, the optimal solution may not be found because the movements of the particles are limited. Large values, on the other hand, can result in particles outmatching good solutions, particles can go beyond a good solution because they move erratically.

The restriction to a range of the velocity is made in order to damp the oscillations deriving from the fact that velocity is a stochastic variable. Setting the upper and lower levels $[-V_{max}, +V_{max}]$, the oscillations of the particles remain inside the allowable range. A formula for the maximum velocity is presented in [30], to ensure uniform velocity throughout all dimensions:

$$V_{max} = \frac{x_{max} - x_{min}}{N}$$

where N is the number of intervals in the k -th dimension selected by the user and x_{max} , x_{min} are maximum and minimum values found so far by the particles.

2.3.4 Modifications of the PSO

Even if the PSO algorithm finds good solutions in less time than the other algorithms, it has some defects. The settings of the acceleration parameters and of the maximum velocity plays a crucial part on the efficiency of the algorithms, and no rules of thumb are known. Empirical studies have shown that, for different values of the parameters, the particles may still diverge leading to the explosion of the swarm. Another drawback that can affect the PSO is the contrary of what just described, it can be affected by premature convergence. The premature convergence causes the particles to be trapped into local optimum and therefore they are not exploring other promising areas of the search space. This leads to the failure of the algorithm specially when the problem is complex. The global maximum may not be found. In order to overcome these problems, different versions of the equation (2.2) has been proposed by different approaches.

Inertia Weight

This approach was introduced by Shi and Eberhart in [32]. They developed this approach to better control the scope of the search and to reduce the importance of V_{max} . The authors starts from the conjecture that: "there is a tradeoff between the

local and global search. For different problems, there should be different balances between the local search ability and global search ability". Considering this, the original algorithm is modified by introducing a new parameter: the inertia weight w . The role of w is to balance the global search and local search, controlling the exploration and exploitation ability of the swarm. The equation to update velocity and position of each particle with the Inertia Weight Approach is:

$$\begin{cases} v_i^{k+1} = w^k v_i^k + U(0, \phi_1) \otimes (p_i^k - x_i^k) + U(0, \phi_2) \otimes (p_g - x_i^k) \\ x_i^{k+1} = x_i^k + v_i^{k+1} \end{cases}$$

The inertia weight controls the exploration of the search space and it can be implemented as a fixed value or can be dynamically changing. The result found is that the best performance can be obtained by a linearly decreasing inertia weight. A high value to w , usually 0.9, is associated initially. This high value allows to find the global optimum neighborhood fast since the particles are moving freely. When the most promising area of the search space is found, the value of w can be decreased, typically to 0.4. Doing so, the search is narrowed and the swarm is no longer in exploratory mode, but it is in exploitative mode. As said in [17]: "Typically, the value of w is decreased gradually during the search process, in an effort to encourage more-intensive local search of already discovered good regions, and to help the swarm converge."

In order to compute the value of the inertia weight that decreases linearly with time, the formula is the following:

$$w^k = w_{max} - \frac{w_{max} - w_{min}}{K} \times k$$

where:

- w_{max} and w_{min} are the initial and final inertia weight values;
- K is the maximum number of iterations of the PSO algorithm;
- $k \in \{1, \dots, K\}$ is the current iteration number.

This approach is very useful in ensuring that the particles of the swarm will converge, but it has some drawbacks. The main disadvantage is that the change of mode (from exploratory to exploitative) is irreversible, when it is made the swarm loses the ability to search in new areas again.

The value of the inertia weight can be adjusted adopting strategies different from the time-decreasing one. An example of a successful strategy is the adaptation of w using a fuzzy system. Through this strategy the performance of the PSO is

improved significantly. Another alternative strategy consists in using an inertia weight with a random component. Eberhart and Shi successfully used $w = U(0.5, 1)$.

Constriction Coefficients

This approach was firstly developed by Clerc and Kennedy in [33]. They introduced a new factor, the constriction coefficient called X , to control the explosion of the swarm and to increase the optimization power of the particle swarm. As we will see from the new velocity update formula, X affects all the components of the right part of the equation, not just the first component as the inertia weight approach does. Equation (2.2) is modified in the following way:

$$\begin{cases} v_i^{k+1} = X \left(v_i^k + U(0, \phi_1) \otimes (p_i^k - x_i^k) + U(0, \phi_2) \otimes (p_g - x_i^k) \right) \\ x_i^{k+1} = x_i^k + v_i^{k+1} \end{cases}$$

where:

$$X = \frac{2}{|2 - \phi - \sqrt{\phi^2 - 4\phi}|} \quad \text{and} \quad \phi_1 + \phi_2 = \phi > 4.0$$

When this approach is used, the value of ϕ is commonly set to 4.1. Considering that $\phi_1 = \phi_2$, the value of the constriction coefficient X is approximately 0.7298. This results in the previous velocity being multiplied by 0.7298 and each of both, the cognitive and the social component, being multiplied by $0.7298 \times 2.05 \approx 1.49618$, since $\phi_1 = \phi_2 = 2.05$ when $\phi = 4.1$ and $X = 0.7298$.

Note that a PSO with constriction coefficient and a PSO with inertia weight are algebraically equivalent in the special case when the optimal setting proposed by Clerc and Kennedy corresponds to $w = 0.7298$ and for the case of inertia weight $\phi_1 = \phi_2 = 1.49618$.

In general, once a particle is focused on the best point in an optimal region, the constriction coefficient improves the convergence of the particle over time, and it does so by reducing the oscillations. The major drawback of this strategy is the fact that particles may follow wider cycles and fail to converge when the individual best performance and the neighbourhood's best performance are in two different regions.

Fully Informed Particle Swarm

If we take into consideration the original version of the PSO, we can notice that the sources that effectively influence each particle are just two: personal best and global best. The particles do not use information concerning the remaining

neighborhoods. To overcome this limitation, Mendes created this new approach, the fully informed particle swarm, in which each particle interact with their neighbors. As said in [29]: "whereas in the traditional algorithm each particle is affected by its own previous performance and the single best success found in its neighborhood, in Mendes' fully informed particle swarm, the particle is affected by all its neighbors, sometimes with no influence from its own previous success". The equation for velocity and position update for this approach is the following:

$$\begin{cases} v_i^{t+1} = X \left(v_i^t + \frac{1}{K_i} \sum_{n=1}^{K_i} U^n \otimes (p_{nbr_k^t} - x_i^t) \right) \\ x_i^{k+1} = x_i^k + v_i^{k+1} \end{cases}$$

where:

- K_i is the number of neighbors for the i -th particle;
- nbr_k is the i 's k -th neighbor.

This approach has proven to find better solutions in fewer iterations than the original version of the algorithm, but its main drawback is the fact that it is much more dependent on the population topology, that is, how particles are connetted to their neighborhood.

2.3.5 Population Topology

In this section I am going to explain the concept of topology of the particle swarm, in order to better understand how the particles are connected and interact with each other. A crucial feature in PSO is the social interaction among the particles of the swarm, they exchange with each other information about the best positions they have found. When a better position is found by a particle, the entire swarm moves in that direction. The importance of social interaction increases the importance of the particles' neighborhood, since the latter determines the extent of social interaction within the swarm. When the neighborhoods of the swarm are large, it is possible to arrive at the convergence of the particles faster, the risk though is that it may be too fast. In the case of small neighborhoods, on the other hand, particles will converge slower because there are less interacrions, but the quality of the solution may be improved.

The general types of neighborhoods in which particles have been studied are essentially two: the global best *gbest* and the local best *lbest*. The original version of the PSO uses the global best population topology. In this topology, the source of social influence on each particle is the best-performing individual in the entire

population. Every particle is connected to every other particle and every particle has access to all the information of the swarm. The particles are attracted to the best solution found by any member of the swarm. To better visualise it, it can be compared to a social network where each individual is connected to any other individual. The *gbest* can be defined as a static topology because neighborhoods and neighbors do not change.

The other common population topology is the local best, *lbest*. It differs from the global best one because particles are no longer all interconnected with each other, but each particle is connected to its immediate neighbors in the population array. This means that each particle has access to the information of its immediate neighbors, and it influences and it is influenced by them. With this topology, particles can be divided in subpopulations. The particles in a subpopulation influence themselves in order to search in their region a local maximum. Thus, subpopulations can search in different regions simultaneously. In figure 2.6 it is possible to see a graphical representation of these two population topologies.

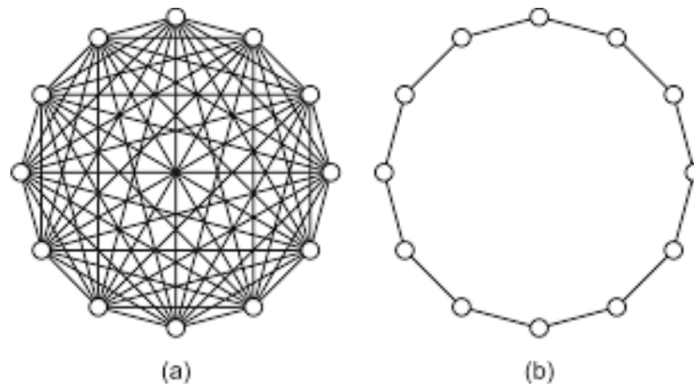


Figure 2.6: *gbest* and *lbest* population topology

There exists a lot of other types of population topology. According to Kennedy and Mendes, one that is worth mentioning is the von Neumann one. With this approach the particles are arranged in a rectangular structure. For example, for a population of twenty individuals, the rectangular structure will be a 5×4 matrix and each particle is connected to four other particles, the one above, below and at each side of it, wrapping the edges. In their work, Kennedy and Mendes, suggested that the von Neumann topology performs better than other topologies, included the *gbest* one. Nevertheless, the population topology is problem specific, it is better to select the most efficient neighborhood on the basis of the problem that has to be solved. An example of the von Neumann population topology can be seen in figure 2.7.

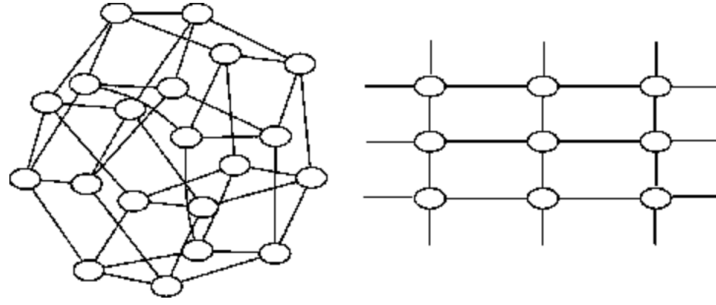


Figure 2.7: von Neumann population topology

2.4 Comparison between GA and PSO

In this section I am going to synthetically compare some general aspects of the two metaheuristics discussed so far. Starting from their creation, GAs are among the first metaheuristics that has been developed, that is in 1960s. PSO, on the other hand, was introduced decades later, in 1995. This suggest that GAs have more studies underneath, while PSO is still be studied and evolved. Another difference is the biological metaphor they are drawn from. GAs derive from an evolutionary metaphor and they are based on genetics and natural selection. PSO, instead, is drawn from a swarm metaphor that has its basis on social concepts and swarm intelligence principles.

Looking more in detail, is possible to see that some of the basic components of GA and PSO are analogous. Particles in PSO and chromosomes in GA, both represents a candidate solution to the problem being addressed.

GAs works using three main operators: selection, crossover and mutation. PSO does not give the same names to its operators but some similarities can be found. Let's consider first crossover. PSO does not have such operator, but its concepts are present, because each particle is stochastically accelerated to its previous best position and to the global best position. This is comparable to the effect of crossover in GAs, it is able to move a chromosome in the problem space. Mutation, instead, for GAs represents the theoretical possibility for a chromosome to reach any point in the search space. In PSO, it is difficult that a particle reaches any point in the search space in one iteration but, given enough iteration, since particles survive iterations, they can eventually go anywhere (given a sufficiently large value of V_{max}). Regarding selection, in GAs consists in the survival of the fittest. The selection operator is not present in PSO, no particle is removed from the population due to survival of the fittest. The particles, though, can change from iteration to iteration due to random velocity and position, and this change can affect almost all the particles.

After having described the metaheuristics that I'm going to use to optimize the portfolio, I will describe in the next chapter the measure of risk and the portfolio that I will use for my analysis.

Chapter 3

Risk Measure and Portfolio Selection Model

In this chapter, I am going to analyze in detail the measure of risk that I am going to use for the analysis. I have decided to use the two-sided coherent risk measure proposed by Chen and Wang [15], introduced in the first chapter, because it is the only one that considers both positive and negative deviations from the expected return simultaneously. Furthermore, with this risk measure it is possible to reflect the investor's risk attitude and to take into consideration the phenomena of asymmetry and fat tails of the distribution. Going on with the chapter, I will introduce the portfolio model that will be applied. The portfolio selection model will be a realistic one, that has been proposed by Corazza, Fasano and Gusso [16].

3.1 Two-sided coherent risk measure in detail

As described in chapter 1, the two-sided coherent risk measure presents more advantages compared to other coherent risk measures. In this section I will focus on its properties.

Let's consider a one-period framework, that starts from current date 0 until a future date D . Risk will be represented by the random payoff Y of some assets or portfolio at time D , defined on a probability space (Ω, \mathcal{F}, P) . The investor will have a random profit when $Y \geq 0$ and a random loss in the case when $Y < 0$. The risk measure $\rho(Y)$ can be considered as the minimum extra cash added to Y that makes the position acceptable for the holder. This means that:

- when $\rho(Y) > 0$, the investor has to add an extra amount of cash of $\rho(Y)$ to ensure the acceptance of his future position;

- when $\rho(Y) < 0$, the investor can withdraw an amount of cash of $\rho(Y)$ without affecting the acceptance of his future position.

Let's define $\|Y\|_p = (E[|Y|^p])^{1/p}$, where $E[\cdot]$ is the expected value of a random variable where $p \in [1, +\infty)$ represents the negative risk. Let Y^- denote $\max(-Y, 0)$ and Y^+ denote $(-Y)^-$. Lastly, define $\sigma_p^\pm(Y) = \|(Y - E[Y])^\pm\|_p$. As said before, this risk measure takes into consideration both sides of the return distribution. The downside of Y is represented by the random variable $(Y - E[Y])^-$, while the upside of Y is represented by the random variable $(Y - E[Y])^+$. Using another notation, the risk measure $\rho_{a,p}(Y)$ considers the lower partial moments, or bad volatility, $\sigma_p^-(Y)$, and the upper partial moments, or good volatility, $\sigma_1^+(Y)$. This two-sided coherent risk measure is then defined by [15] as follows: given $p \in [1, \infty)$, $a \in [0, 1]$, the new risk measure $\rho_{a,p} : L^p(Q) \rightarrow \mathbf{R}$ is determined by:

$$\begin{aligned} \rho_{a,p}(Y) &= a\sigma_1^+(Y) + (1-a)\sigma_p^-(Y) - E[Y] \\ &= a\|(Y - E[Y])^+\|_1 + (1-a)\|(Y - E[Y])^-\|_p - E[Y] \end{aligned} \quad (3.1)$$

where:

- a is a global risk factor that serves to adjust the balance between good volatility and bad volatility. Setting different values, it can also represent risk neutrality or risk preference. Risk neutrality by setting $p = 1$ and $a = 0.5$, and risk preference by always setting $p = 1$ but changing the value of a : $0.5 < a \leq 1$. In general, the smaller the value of a , the bigger the weight on the lower p -th partial moment, that corresponds to a more cautious investor;
- p is the the local risk factor that controls the investment. The more risk averse is the investor, the greater the value of p . Choosing suitable values of p , it is possible to account the skewness and/or leptokurtosis of the return distribution.

$\rho_{a,p}(Y)$ is generated by firstly taking the 1-norm of the positive deviation $(Y - E[Y])^+$, and the p -norm of the negative deviation $(Y - E[Y])^-$, and then taking the convex combination of these two norms. Another difference from other risk measures is that equation (3.1) includes simultaneously $\|(Y - E[Y])^+\|_1$ and $E[Y]$. The term $E[Y]$ within the equation ensures that the risk measure $\rho_{a,p}(Y)$ satisfies the properties of coherence. In particular, the following theorems hold:

Theorem 3.1 For any $p : 1 \leq p \leq \infty$ and $a \in [0, 1]$, the risk measure $\rho_{a,p}$

defined by (3.1) is a coherent risk measure.

Theorem 3.2 The coherent risk measure $\rho_{a,p}$ is non-decreasing with respect to p , and is non-increasing with respect to a , respectively.

Theorem 3.2 is called the monotonic property of $\rho_{a,p}$ and it reflects the investor's attitude toward risk. More precisely, the non-decreasing property of this risk measure with respect to p means that the investor, using $\rho_{a,p}$ as measure of risk, is going to consider Y riskier when, fixed the value of a , the value of p is higher. Consequently, when the value of p is high, $\rho_{a,p}$ will be larger than when an investor adopts the same risk measure with a lower value of p . Considering that p is the parameter that refers just to the negative tail, and so to the risk, the negative variance, a large value of p represents an investor that is more careful to negative risk. As concerns the non-increasing property of $\rho_{a,p}$ with respect to a , it means that when the value of p is fixed, the risk measure reflects the less risk-averse investor when the value of a is large.

The main characteristic that differs this two-sided risk measure from all other one-sided risk measures is the fact that since $E((Y - E[Y])^+) = E((Y - E[Y])^-)$, when the investor minimizes $\|(Y - E[Y])^+\| = E((Y - E[Y])^+)$, he is minimizing his dispersion from the mean. The most important advantage that is possible to have with $\rho_{a,p}(Y)$ is that it can be used to find optimal portfolios that take into consideration reality conditions more than those under CVaR.

3.2 Portfolio Selection Model

Now that an appropriate measure of risk to be minimized is decided, it is important to set some constraints in order to create a realistic portfolio selection model. For the purposes of my analysis, I am going to use the model proposed by Corazza, Fasano and Gusso in [16]. It is important to note that making effective portfolio selection in real stock markets is not easy for different reasons. The first reason concerns the selection of the risk measure, that should satisfy the properties of coherence and represent the non normal distribution of returns. This is satisfied by the two-sided coherent risk measure $\rho_{a,p}$. The second reason is the fact that it is important to take into account rules of the portfolio management industry, such as bounds on the number of stocks to trade, that can affect the portfolio selection process. This can be done by introducing some constraints in the problem. The third reason is that using jointly the measure of risk and rules of the portfolio management industry (formalized in terms of constraints), the portfolio selection

problem created is possibly highly nonlinear, nondifferentiable, nonconvex and mixed-integer. In other words, the problem to solve is a NP-hard problem. In order to provide a "cheap and reliable" solution, an exact penalty method is used, combined with the two metaheuristics taken into consideration, first with PSO and secondly with GAs. The exact penalty scheme is used because it transforms the nonlinear, nondifferentiable and mixed-integer portfolio selection problem (that results from taking into consideration the risk measure and the constraints), into an equivalent, in term of solutions, minimization problem that is unconstrained. The resulting problem is still nonlinear, nondifferentiable and non convex. To approximately compute a global minimizer of the exact penalty-based model it is necessary to use an algorithm that does not use derivatives; this is provided by GAs and PSO.

Now I will describe all the constraints that are going to form the portfolio selection model.

Return and budget constraints

The return and budget constraints are included also in the Markowitz portfolio model and are the most important, since they are essential to form the portfolio. Let's start with some definitions. Suppose there are N assets to choose from, and for $i = 1, \dots, N$, $x_i \in \mathbb{R}$ represents the weight of the i -th asset included in the portfolio, with $X^T = (x_1, \dots, x_N)$, and r_i is a real value random variable that represents the return of the i -th asset. The expected value of r_i , $E[r_i]$, is represented by \hat{r}_i . Then, the return of the whole portfolio $R \in \mathbb{R}$ and its expected value \hat{R} can be expressed as follows:

$$R = \sum_{i=1}^N x_i r_i$$

$$\hat{R} = \sum_{i=1}^N x_i \hat{r}_i$$

The return constraint is then defined by:

$$\hat{R} \geq l, \quad \text{with } l > 0$$

where l is the minimum expected return that is desired from the portfolio. This constraint allows to select the portfolio among the ones that, minimizing the measure of risk, lies on the efficient frontier.

The budget constraint can be represented as follows:

$$\sum_{i=1}^N x_i = 1.$$

This constraint ensures that all the capital available is invested.

Cardinality constraint

The use of a cardinality constraint in the portfolio selection model brings several advantages. It serves to limit the number of assets that a portfolio can have, and it also limits the proportion of the portfolio held in a given asset. Doing so it allows to select a not too small or too large subset of the available assets. This leads to an indirect control of transaction costs, because this constraint helps fund managers with the problem of building a portfolio by choosing from hundreds of assets. In fact, a too large number of assets can involve accounting problems that can increase transaction costs.

Let $Z^T = (z_1, \dots, z_N) \in \{0, 1\}^N$ be a binary vector, such that $z_i = 1$ if the i -th asset is included in the portfolio, and $z_i = 0$ if it is not included in the portfolio. The cardinality constraint can be defined by:

$$K_d \leq \sum_{i=1}^N z_i \leq K_u, \quad \text{where } 1 \leq K_d \leq K_u \leq N$$

where K_d and K_u represents respectively the minimum and the maximum number of assets included in the portfolio.

It is also important to require that each of the selected assets can not be a too small or too large fraction of the portfolio. This can be done by establishing a minimum, d , and a maximum, u , fraction allowed to allocate in each asset:

$$z_i d \leq x_i \leq z_i u \quad \text{where } 0 \leq d \leq u \leq 1.$$

To conclude, in order to ensure compatibility between the cardinality constraint and the establishment of the minimum and maximum fractions, parameters d and u must satisfy:

$$d \leq \frac{1}{K_d} \quad \text{and} \quad u \geq \frac{1}{K_u}$$

With the use of these constraints, the portfolio is more realistic with respect to the Markowitz portfolio model. Following, I will describe the portfolio selection model that I am going to use.

Portfolio Selection Model

The portfolio selection model is now composed by putting together all the previously described constraints. The goal of the model is to minimize $\rho_{a,p}$. The resulting portfolio selection problem is expressed as follows:

$$\text{minimize}_{X,Z} \rho_{a,p}(R)$$

$$\text{subject to: } \hat{R} \geq l$$

$$\sum_{i=1}^N x_i = 1$$

$$K_d \leq \sum_{i=1}^N z_i \leq K_u$$

$$z_i d \leq x_i \leq z_i u, \quad \text{with } i = 1, \dots, N$$

$$z_i \in \{0, 1\}, \quad \text{with } i = 1, \dots, N$$

Penalty function method

Now it is necessary to reformulate the problem, in order for it to become an unconstrained problem. Through the use of a penalty method it is possible to approximate a constrained optimization problem to an unconstrained one. This approximation is made by adding to the objective function a term that prescribes a high cost for the violation of the constraints. The severity of the penalty that will be applied if the constraints are violated is determined by a parameter, called ϵ .

There are two main issues regarding the penalty function:

- How well the unconstrained problem approximates the constrained problem. The degree of approximation is given by the parameter ϵ , and as it tends to infinity, the solution of the unconstrained problem converges to the solution of the constrained one;
- How to solve the unconstrained problem, when its objective function is containing the penalty term. As the value of ϵ increases, the approximation of the problem increases, but also the structure of the unconstrained problem becomes increasingly unfavorable. This can lead to a slower rate of convergence, meaning that finding a good solution requires more time.

An important advantage deriving from the use of a penalty method is that it is possible to solve constrained problems without the need of sophisticated algorithms.

As described in [38], penalty methods can be described as following. Consider the problem:

$$\begin{aligned} & \text{minimize } f(x) \\ & \text{subject to } h_j(x) = 0, \quad g_i(x) \leq 0 \end{aligned}$$

where f is a continuous function on \mathbb{R}^n , $h_j(x)$, with $j = 1, \dots, m$, is a set of m constraints expressed in equality form, and $g_i(x)$, with $i = 1, \dots, p$, is a set of p constraints expressed in inequality form. The penalty function method consists in replacing this problem with another one, not subject to constraints, as follows:

$$\text{minimize } f(x) + \epsilon P(x)$$

where ϵ is a positive constant and P is a function on \mathbb{R}^n that satisfies the following properties:

- P is continuous;
- $P(x) \geq 0$ for all $x \in \mathbb{R}^n$;
- $P(x) = 0$ if and only if $x \in S$.

The absolute-value penalty function is expressed as:

$$P(x) = \sum_{j=1}^m |h_j(x)| + \sum_{i=1}^p \max(0, g_i(x)).$$

For our portfolio we consider an exact penalty method, "where the term exact refers to the correspondence between the minimizers of the original constrained problem and the minimizers of the unconstrained (penalized) one" [16]. This means that the solution of the penalty problem is equal to the solution of the constrained original problem. This is guaranteed by the Exact Penalty Theorem [38]:

Theorem 3.3 "Suppose that the point x^* satisfies the second-order sufficiency conditions for a local minimum of the constrained problem. Let λ and μ be the corresponding Lagrange multipliers. Then for $\epsilon > \max\{|\lambda_i|, \mu_j : i = 1, \dots, m \quad j = 1, \dots, p\}$, x^* is also a local minimum of the absolute-value penalty objective."

This theorem provides the conditions to make sure that the solutions of the con-

strained and unconstrained problem coincides. Note that, in the theorem, there is no indication of what the value of the parameter ϵ should be.

In the next section, the penalty function is considered and the portfolio, as a consequence, will be reformulated.

Portfolio Selection Model with Exact Penalty Method

Before building the portfolio selection model with the exact penalty function, it is necessary to rewrite the constraints of the original portfolio selection model previously described in this chapter. This is necessary because the penalty function is applied each time a constraint is violated. Each constraint can be reformulated in the following way:

- $\hat{R} \geq l \rightarrow \max\{0, l - \hat{R}\} = 0$
- $\sum_{i=1}^N x_i = 1 \rightarrow \left| \sum_{i=1}^N x_i - 1 \right| = 0$
- $K_d \leq \sum_{i=1}^N z_i \rightarrow \max\{0, K_d - \sum_{i=1}^N z_i\} = 0$
- $\sum_{i=1}^N z_i \leq K_u \rightarrow \max\{0, \sum_{i=1}^N z_i - K_u\} = 0$
- $z_i d \leq x_i \rightarrow \sum_{i=1}^N \max\{0, z_i d - x_i\} = 0$
- $x_i \leq z_i u \rightarrow \sum_{i=1}^N \max\{0, x_i - z_i u\} = 0$
- $z_i \in \{0, 1\} \rightarrow \sum_{i=1}^N |z_i(1 - z_i)| = 0$

The portfolio selection model has become as follows:

$$\text{minimize}_{X,Z} P(X, Z; \epsilon)$$

where:

$$\begin{aligned} P(X, Z; \epsilon) = & \rho_{a,p}(R) + \frac{1}{\epsilon} \left[\max\{0, l - \hat{R}\} + \left| \sum_{i=1}^N x_i - 1 \right| + \max\left\{0, K_d - \sum_{i=1}^N z_i\right\} \right. \\ & + \max\left\{0, \sum_{i=1}^N z_i - K_u\right\} + \sum_{i=1}^N \max\{0, z_i d - x_i\} \\ & \left. + \sum_{i=1}^N \max\{0, x_i - z_i u\} + \sum_{i=1}^N |z_i(1 - z_i)| \right] \end{aligned}$$

where ϵ is the penalty parameter.

Following the same idea, it is also necessary to rewrite the measure of risk for any portfolio $X = (x_1, \dots, x_N)^T$ as:

$$\rho_{a,p}(R) = \frac{a}{T} \left[\sum_{t=1}^T \left(\sum_{i=1}^N (r_{i,t} - \hat{r}_i) x_i \right)^+ \right] + (1-a) \left\{ \frac{1}{T} \sum_{t=1}^T \left[\left(\sum_{i=1}^N (r_{i,t} - \hat{r}_i) x_i \right)^- \right]^p \right\}^{\frac{1}{p}}$$

where:

- $r_{i,t}$ is the return of the stock and is calculated using the price series:

$$r_{i,t} = \frac{p_{i,t+1} - p_{i,t}}{p_{i,t}};$$

- \hat{r}_i is the expected value of r_i and is estimated using historical data:

$$\hat{r}_i = \frac{1}{T} \sum_{t=1}^T r_{i,t}.$$

Now that the portfolio selection model is an unconstrained one, and it has been reformulated, it is possible to proceed with the analysis and apply to it the two metaheuristic methods taken into consideration: GAs and PSO.

Chapter 4

Applications to the FTSE MIB Stock Index

In this chapter the applications of the model and the metaheuristics described so far are described. The solution procedure will be implemented on real data, on the italian stock index FTSE MIB. The time length of the historical data considered is from April 2012 to March 2018.

4.1 Problem Setting

This analysis considers the daily closing prices of the assets contained in the FTSE MIB index. The FTSE MIB is the most significant italian stock index, and it is based on the stocks of 40 companies. The companies whose securities are in the index are the one that have the highest values on capitalization, liquidity and number of stocks issued. The data considered are the daily closing prices from April 2012 until March 2018. For the purpose of this analysis only 31 securities are considered, because 9 stocks of the index had missing data in the period requested. The securities considered are listed in Table 4.1.

In this analysis the whole period has been divided into eight subsets. This 8 time periods have length of 9 months and each period is then divided in two parts: 6 months of in-sample and 3 months of out-of-sample. The in-sample period serves to estimate the parameters of the model. The out-of-sample period, is the investment period, the virtual future, and serves to verify the effectiveness of the model. The idea is that the portfolio risk and expected return estimated in the first period will remain, more or less, the same also in the investement period. This means that the percentage of investment suggested by the in-sample analysis should provide the best portfolio also for the near virtual future. The eight time periods are divided

A2A S.p.A.	Mediaset S.p.A.
Atlantia S.p.A.	Mediobanca S.p.A.
Azimut Holding S.p.A.	Prysmian S.p.A.
Banca Generali S.p.A.	Recordati S.p.A.
Banco BPM S.p.A.	Saipem S.p.A.
BPER Banca S.p.A.	Salvatore Ferragamo S.p.A.
Brembo S.p.A.	Snam S.p.A.
Buzzi Unicem S.p.A.	STMicroelectronics N.V.
Campari - Milano S.p.A.	Telecom Italia S.p.A.
Enel S.p.A.	Tenaris S.A.
Eni S.p.A.	Terna - Rete Elettrica Nazionale S.p.A.
Exor N.V.	UBI Banca S.p.A.
Generali S.p.A.	UniCredit S.p.A.
Intesa Sanpaolo S.p.A.	Unipol S.p.A.
Leonardo S.p.A.	UnipolSai Assicurazioni S.p.A.
Luxottica S.p.A.	

Table 4.1: FISE MIB stocks

as follows:

- Period 1: in-sample: April 2012 - September 2012
out-of-sample: October 2012 - December 2012;
- Period 2: in-sample: January 2013 - June 2013
out-of-sample: July 2013 - September 2013;
- Period 3: in-sample: October 2013 - March 2014
out-of-sample: April 2014 - June 2014;
- Period 4: in-sample: July 2014 - December 2014
out-of-sample: January 2015 - March 2015;
- Period 5: in-sample: April 2015 - September 2015
out-of-sample: October 2015 - December 2015;
- Period 6: in-sample: January 2016 - June 2016
out-of-sample: July 2016 - September 2016;
- Period 7: in-sample: October 2016 - March 2017
out-of-sample: April 2017 - June 2017;
- Period 8: in-sample: July 2017 - December 2017
out-of-sample: January 2018 - March 2018.

The daily closing prices have been used to compute the daily return of each stock, using the following formula:

$$r_{i,t} = \frac{p_{i,t+1} - p_{i,t}}{p_{i,t}},$$

and its expected value has been estimated using historical data, through the formula:

$$\hat{r}_i = \frac{1}{T} \sum_{t=1}^T r_{i,t}$$

where $t = 1, \dots, T$ and T is the time horizon considered, and $i = 1, \dots, 31$ is the asset considered.

The algorithms used were launched on MATLAB R2018a and the experiment has been performed on a MacBook Pro with a Intel Core i5 processor, with 8gb ram.

Problem specific parameter setting

In this section the parameters related to the constraints of the portfolio selection problem are set. Note that this parameters are equal for both algorithms, PSO and GAs. The parameters setted are listed following:

- Minimum expected return desired from the portfolio: $l = 2\%$;
- Minimum fraction allowed in each asset: $d = 2\%$;
- Maximum fraction allowed in each asset: $u = 20\%$;
- Minimum number of holding assets: $K_d = 5$;
- Maximum number of holding assets, two different values are set: $K_u = 10$ and $K_u = 20$;
- Global risk factor in the risk measure: $a = 0.5$;
- Local risk factor in the risk measure: $p = 2$.

PSO parameters setting

The approach used for the PSO in this analysis is the inertia weight one. The parametes are set as suggested by the literature:

- Inertia weight: $\omega = 0.7298$;
- Cognitive acceleration coefficient: $\phi_1 = 1.49681$;

- Social acceleration coefficient: $\phi_2 = 1.48961$;
- Number of iterations: 1000.

In order to determine the values of the penalty parameter ϵ and the number P of particles in the swarm, some preliminary tests have been performed. It has been decided to use Period 1 as representative of the whole period to run the preliminary tests. To assess the optimal value of the penalty parameter, the algorithm has run 10 times for different values of ϵ , considering $P = 62$ number of particles in the swarm. The number of particles has been temporarily set equal to 62 because it is the minimum number of particles that can be used in this problem, since there are 62 variables: 31 x , the securities available, and 31 z , a binary variable that is going to indicate if a security is in the portfolio or not. Once the penalty parameter is set, the next step will be to test if $P = 62$ particles in the swarm is the best value to use or not, but firstly it is necessary to set the value of ϵ , since it assures the correspondence between the original constrained problem and the unconstrained one that is going to be solved. To determine which value is better in this analysis, the average best value of the fitness function normalized and its standard deviation has been taken into account. In Table 4.2 are reported the results. As suggested by Table 4.2, the optimal value of ϵ is 0.000001, since its standard deviation is the lowest.

ϵ	Normalized Fitness	Standard Deviation
0.0001	0.500345153	0.283430188
0.00001	0.532340790	0.304006940
0.000001	0.581226326	0.256011254
0.0000001	0.464502713	0.279978117
0.00000001	0.520694327	0.348438354

Table 4.2: Results for different choices of the parameter ϵ .

To decide the number of particles to use for the analysis, tests have been made for different values of P . Note that always multiples of the number of securities in the portfolio have been considered. The algorithm has run 10 times for different values of P using the value of ϵ suggested by the previous table. The results are shown in Table 4.3.

As before, the normalized fitness function and its standard deviation have been taken into consideration to take the decision. The table suggests to use $P = 62$ number of particles, since its standard deviation is the lowest.

P	Normlized Fitness	Standard Deviation
62	0.581226326	0.256011254
93	0.596409271	0.266367739
124	0.582205015	0.269504334

Table 4.3: Results for different choices of the number P of particles in the PSO.

GAs parameters settings

For the analysis regarding GAs, it has been decided to use the application provided by Matlab "Global Optimization Toolbox". The aim of the analysis is to verify if one metaheuristic provides better results with respect to the other, *ex ceteris paribus*. In order to have a consistent comparison the same values have been used, where possible. The size of the population considered is the same, 62 individuals. Also the number of iteration is not changed: 1000 iterations. The penalty parameter $\epsilon = 0.000001$ tested and used in PSO is kept equal. Regarding GAs specific parameters the setting is the following:

- *Selection Operator:*

The selection operator used belong to the family of Fitness Proportionate Selection. In particular, in this algorithm the roulette-wheel selection is applied. As described in the second chapter, with this method each individual has a slot with the size proportional to its fitness;

- *Crossover Operator:*

The crossover operator used is the single-point crossover and the probability applied to crossover is the default one, $p_c = 0.8$. It has been decided to use a basic crossover operator because the version of the PSO used is also a basic one, and so the conditions in which the two metaheuristics work are the same;

- *Mutation Operator:*

The value of the probability of mutation used is $p_m = 0.01$.

4.2 Application and discussion

After setting the parameters needed and running the preliminary tests, let's start with the analysis of the data. The aim of this analysis is to test the capabilities of PSO and GAs to find a global minimum for the optimization problem and look how effective the portfolio obtained is. This is made considering different risk preferences of the possible investor.

The first part of the analysis is made using PSO. I started analyzing each of the eight periods individually. To each period different values of K_u (the maximum number of holding assets) are applied: first a value of 10 and then a value of 20. For each value of K_u a different value of the parameter p is applied. As said in the previous chapter, p is the parameter of the risk measure that refers to the negative variance. A small value of p indicates an investor that is less concerned about negative risk and thus is risk-seeker. A high value of p , on the contrary, indicates an investor that is more careful about negative risk, more risk-averse. Three different values are taken into consideration: $p = 1$, $p = 2$, $p = 5$. Regarding the other parameter of the risk measure a , that is the global risk factor that balances good and bad volatility, its value is fixed to $a = 0.5$. For each different value of K_u and p , the algorithm run 10 times. This because metaheuristics are not an exact science, so a run could be particularly good or particularly bad. In Table 4.4 and 4.5 the results are shown.

Period 1	p=1	p=2	p=5
$\rho_{0.5,p}; K_u = 10$	0.046285115	0.101275168	0.301566992
$N_{a,p}(K_u)$	15	17	16
$\rho_{0.5,p}; K_u = 20$	0.022570107	0.050308516	0.197466737
$N_{a,p}(K_u)$	15	15	16
Period 2	p=1	p=2	p=3
$\rho_{0.5,p}; K_u = 10$	0.008563362	0.014435054	0.023255508
$N_{a,p}(K_u)$	18	15	15
$\rho_{0.5,p}; K_u = 20$	0.005471717	0.008697262	0.013214048
$N_{a,p}(K_u)$	14	17	17
Period 3	p=1	p=2	p=3
$\rho_{0.5,p}; K_u = 10$	0.008890015	0.012075873	0.021815713
$N_{a,p}(K_u)$	14	15	15
$\rho_{0.5,p}; K_u = 20$	0.004385419	0.006049077	0.01087941
$N_{a,p}(K_u)$	16	16	14

Table 4.4: Monotonicity of $\rho_{a,p}$ for $a = 0.5$ and different values of p and K_u , with six months data for period 1, 2, 3. $N_{a,p}(K_u)$ represents the number of assets in the final portfolio.

Period 4	p=1	p=2	p=3
$\rho_{0.5,p}; K_u = 10$	0.00794993	0.010697401	0.017934831
$N_{a,p}(K_u)$	15	16	15
$\rho_{0.5,p}; K_u = 20$	0.005157948	0.007198964	0.01046431
$N_{a,p}(K_u)$	15	17	15
Period 5	p=1	p=2	p=3
$\rho_{0.5,p}; K_u = 10$	0.008782711	0.013049929	0.019695973
$N_{a,p}(K_u)$	15	16	16
$\rho_{0.5,p}; K_u = 20$	0.00572847	0.008713064	0.014171558
$N_{a,p}(K_u)$	16	16	15
Period 6	p=1	p=2	p=3
$\rho_{0.5,p}; K_u = 10$	0.024331483	0.061563977	0.140302141
$N_{a,p}(K_u)$	15	16	14
$\rho_{0.5,p}; K_u = 20$	0.010085331	0.016630463	0.075221928
$N_{a,p}(K_u)$	15	15	14
Period 7	p=1	p=2	p=3
$\rho_{0.5,p}; K_u = 10$	0.00964164	0.012764995	0.023796341
$N_{a,p}(K_u)$	15	16	16
$\rho_{0.5,p}; K_u = 20$	0.005207038	0.007332092	0.01182636
$N_{a,p}(K_u)$	16	15	16
Period 8	p=1	p=2	p=3
$\rho_{0.5,p}; K_u = 10$	0.005700071	0.008505462	0.018645061
$N_{a,p}(K_u)$	15	15	14
$\rho_{0.5,p}; K_u = 20$	0.003292545	0.004525696	0.006622383
$N_{a,p}(K_u)$	15	17	15

Table 4.5: Monotonicity of $\rho_{a,p}$ for $a = 0.5$ and different values of p and K_u , with six months data for period 4, 5, 6, 7, 8. $N_{a,p}(K_u)$ represents the number of assets in the final portfolio.

In Table 4.4 and 4.5 the averages of the 10 runs of the risk measure of the portfolio at its best are reported. This means that the values taken into consideration are the one of the particle that performs better in the whole swarm, so the risk is the minimum risk. In the second and fourth rows, $N_{a,p}(K_u)$ is the number of assets present in the final optimal portfolio. It is important to notice that the monotonicity theorem (Theorem 3.2) is respected. In this table it is possible to see that the risk measure is non-decreasing with respect to p in fact we have that: $\rho_{0.5,1} < \rho_{0.5,2} < \rho_{0.5,5}$ for both values of K_u in each period. In fact, an investor that

is more cautious with respect to negative risk is going to utilize a higher value of p , and thus considers the investment more risky. On the other hand, an investors that seeks more risk is going to use a lower value of p and he is going to perceive the portfolio as less risky.

An interesting thing to look at is the number of assets in the portfolio. In particular, in the case of $K_u = 10$, the cardinality constraint that limits the number of assets in the portfolio is always violated. In fact, the number of assets in the portfolio when $K_u = 10$ range between 14 and 18, that is similar to the case $K_u = 20$, when the maximum number of assets allowed in the portfolio is 20. When $K_u = 20$, the range is between 14 and 17. The violation of the upper bound of the cardinality constraint is not that worrying. Other constraint are much more important to be respected, for example the budget constraint (if the capital invested exceeds the budget, problems may arise). Since there are not other significant constraint violations, I will continue the analysis without worrying about it. A last consideration is made concerning the value of $\rho_{a,p}$ with respect to the two values of K_u . In all the periods the value of the risk measure is higher when the maximum number of assets allowed in the portfolio is 10, and lower when the assets allowed are 20. This means that PSO has performed a good exploration in the set of $K_u = 10$, that is a subset of $K_u = 20$.

Theorem 3.2 states also that $\rho_{a,p}$ is non-increasing with respect to a . In order to prove this, the same computations made for different values of p are made for different values of a . In this case the value of p is held constant, $p = 2$. The different values of a used are: $a = 0, a = 0.25, a = 0.5, a = 0.75, a = 1$. In table 4.6 and 4.7 it is possible to see the results.

Period 1	a=0	a=0.25	a=0.50	a=0.75	a=1
$\rho_{a,2}; K_u = 10$	0.294616277	0.121459337	0.101275168	0.100526273	0.053466708
$N_{a,p}(K_u)$	16	16	17	16	16
$\rho_{a,2}; K_u = 20$	0.056320392	0.051799679	0.050308516	0.045883051	0.042102011
$N_{a,p}(K_u)$	15	16	15	16	15
Period 2	a=0	a=0.25	a=0.50	a=0.75	a=1
$\rho_{a,2}; K_u = 10$	0.020292469	0.016754638	0.014435054	0.011632733	0.009089761
$N_{a,p}(K_u)$	17	14	15	15	15
$\rho_{a,2}; K_u = 20$	0.010631988	0.010241643	0.008697262	0.006900491	0.005270991
$N_{a,p}(K_u)$	14	15	17	15	17

Table 4.6: Monotonicity of $\rho_{a,p}$ for $p = 2$ and different values of a and K_u , with six months data for period 1, 2. $N_{a,p}(K_u)$ represents the number of assets in the final portfolio.

Period 3	a=0	a=0.25	a=0.50	a=0.75	a=1
$\rho_{a,2}; K_u = 10$	0.016904648	0.015111245	0.012075873	0.010760755	0.008328007
$N_{a,p}(K_u)$	16	17	15	17	17
$\rho_{a,2}; K_u = 20$	0.008495001	0.007182894	0.006049077	0.005224888	0.004737236
$N_{a,p}(K_u)$	16	16	16	16	15
Period 4	a=0	a=0.25	a=0.50	a=0.75	a=1
$\rho_{a,2}; K_u = 10$	0.013245515	0.013004366	0.010697401	0.008843301	0.007622232
$N_{a,p}(K_u)$	14	15	16	15	15
$\rho_{a,2}; K_u = 20$	0.009422174	0.008554866	0.007198964	0.005637681	0.004904895
$N_{a,p}(K_u)$	16	15	17	16	15
Period 5	a=0	a=0.25	a=0.50	a=0.75	a=1
$\rho_{a,2}; K_u = 10$	0.015527783	0.013350062	0.013049929	0.010651811	0.007985509
$N_{a,p}(K_u)$	15	15	16	16	15
$\rho_{a,2}; K_u = 20$	0.010989931	0.009710244	0.008713064	0.006982828	0.005738071
$N_{a,p}(K_u)$	14	17	16	16	17
Period 6	a=0	a=0.25	a=0.50	a=0.75	a=1
$\rho_{a,2}; K_u = 10$	0.108021924	0.082038239	0.061563977	0.021655929	0.01327776
$N_{a,p}(K_u)$	15	16	16	15	15
$\rho_{a,2}; K_u = 20$	0.020792105	0.019813861	0.016630463	0.013358931	0.011711872
$N_{a,p}(K_u)$	14	15	15	15	15
Period 7	a=0	a=0.25	a=0.50	a=0.75	a=1
$\rho_{a,2}; K_u = 10$	0.016448314	0.013232943	0.012764995	0.010961346	0.010704217
$N_{a,p}(K_u)$	16	15	16	15	15
$\rho_{a,2}; K_u = 20$	0.009053139	0.00844011	0.007332092	0.005712157	0.004672278
$N_{a,p}(K_u)$	16	17	15	15	14
Period 8	a=0	a=0.25	a=0.50	a=0.75	a=1
$\rho_{a,2}; K_u = 10$	0.010599423	0.010320686	0.008505462	0.006410563	0.005893359
$N_{a,p}(K_u)$	16	17	15	15	16
$\rho_{a,2}; K_u = 20$	0.005891882	0.005882982	0.004525696	0.003591684	0.002889844
$N_{a,p}(K_u)$	15	14	17	15	15

Table 4.7: Monotonicity of $\rho_{a,p}$ for $p = 2$ and different values of a and K_u , with six months data for period 3, 4, 5, 6, 7, 8. $N_{a,p}(K_u)$ represents the number of assets in the final portfolio.

It is possible to see that the non-increasing property of a holds, in fact we have that $\rho_{0,2} > \rho_{0.25,2} > \rho_{0.5,2} > \rho_{0.75,2} > \rho_{1,2}$. As said in the previous chapter, when the value of a is low, the weight on the negative risk is higher. This means that lower values of a reflects an investor that is more careful about negative risk and

perceive the investment as more risky. On the other hand, higher values of a put less weight on the negative variance, reflecting an investor that is more risk-seeker and thus perceive the investment as less risky. Also in this case, the upper bound of the cardinality constraint is violated. The number of assets present in the final portfolio when $K_u = 10$ ranges between 14 and 17, that is the same as when the value of K_u is fixed at 20. Despite this, it is possible to say that PSO performed a good exploration when $K_u = 10$, since $\rho_{a,p}$ is lower when $K_u = 20$ than when $K_u = 10$.

Now, I am going to analyze the performances in the out-of-sample periods of the portfolios selected. For each of the eight periods we have the final portfolio, that is the optimal one found with PSO, that has been evaluated in the six months of in-sample period. To verify the efficiency of this portfolio, it is now invested in the next three months, the out-of-sample period. The percentage returns are computed for each portfolio with different values of K_u and p . As before, the percentage returns are the average of 10 runs. Table 4.8 shows the results.

It is possible to see immediately that the returns are very low. The return constraint that asked the portfolio to have an expected return greater or equal to 2% is almost always violated. Knowing that the period of the data (from april 2012 until march 2018) is a period in which the Italian economy has suffered, this is not an unexpected result. Let's start analyze this results. In the first period, portfolios with $K_u = 20$ performs better that portfolios with $K_u = 10$, in particular the best portfolio for this period is reached with the combination $K_u = 20$ and $p = 1$. In period 2 the best results are also when $K_u = 20$, except for the case in which $p = 1$, where the portfolio with $K_u = 10$ is better. The portfolio that performs better in this period is when $K_u = 20$ and $p = 2$. The third period is the period that has the worst performances with respect to all other periods. In this period the returns when $K_u = 10$ are slightly higher, besides when $p = 1$. The only positive return in this period is when $K_u = 10$ and $p = 2$. Period 4 has higher returns when $K_u = 20$, except made for $p = 5$. The best performance in this period is when $K_u = 10$ and $p = 5$. In period 5, performances are better when $K_u = 10$ except when $p = 5$. The portfolio that performs the best in this period is when $K_u = 10$ and $p = 1$. In period 6 the best performance is when $K_u = 10$ and $p = 2$. For the other two values of p , the higher performance is when $K_u = 20$. Period 7 is the period where there are in absolute the best performances. There are higher returns when $K_u = 10$, except when $p = 1$. The best portfolio performance in this period is in the case when $K_u = 10$ and $p = 5$. Also in the last period, when $K_u = 10$ performances are better, again except when $p = 1$. The best performance in this period, is when $K_u = 10$ and $p = 5$. To sum up, the best performance is provided

from the portfolio with $K_u = 10$ and $p = 5$ in period 7, with a return of 2.07%, while the worst performance is -0.15% and it is in period 3, when $K_u = 10$ and $p = 1$. It is possible to see that in periods 1, 4, 7 and 8 an investor that is more cautious about negative risk will have higher returns with respect to an investor that is more risk-seeker. In these periods, for both values of K_u , the performances when $p = 5$ are higher than when $p = 1$. In period 2 it is the contrary, a less cautious investor will have higher returns, this because portfolios with $p = 1$ provides higher returns than portfolios with $p = 5$.

Period 1	p=1	p=2	p=5
$K_u = 10$	0.07910%	0.02258%	0.10185%
$K_u = 20$	0.14283%	0.06809%	0.11946%
Period 2	p=1	p=2	p=5
$K_u = 10$	0.24083%	0.07027%	0.12622%
$K_u = 20$	0.22341%	0.28541%	0.22816%
Period 3	p=1	p=2	p=5
$K_u = 10$	-0.14813%	0.02289%	-0.13231%
$K_u = 20$	-0.04394%	-0.09841%	-0.14691%
Period 4	p=1	p=2	p=5
$K_u = 10$	0.39491%	0.33666%	0.47080%
$K_u = 20$	0.39973%	0.37244%	0.41671%
Period 5	p=1	p=2	p=5
$K_u = 10$	0.12491%	0.08478%	0.03413%
$K_u = 20$	0.07303%	0.05890%	0.09128%
Period 6	p=1	p=2	p=5
$K_u = 10$	0.06657%	0.16152%	0.02341%
$K_u = 20$	0.09235%	0.07827%	0.10425%
Period 7	p=1	p=2	p=5
$K_u = 10$	0.08676%	1.18580%	2.06798%
$K_u = 20$	1.03901%	0.26565%	1.30446%
Period 8	p=1	p=2	p=5
$K_u = 10$	-0.10758%	-0.01715%	0.09008%
$K_u = 20$	-0.02009%	-0.04166%	0.00299%

Table 4.8: Portfolio out-of-sample returns with different values of p and K_u for all eight time periods.

The next analysis made concerns the number of particles used in the swarm. I decided, as previously described in this chapter, to use $P = 62$ particles since this number of particles leads to a lower standard deviation of the normalized fitness function, and to make the analysis comparable with the one made using GAs. In order to see how well or badly a swarm with 62 particles converges, it is compared with the convergence of two other swarms, one with $P = 93$ particles and the other with $P = 124$ particles. As before, period 1 is taken into consideration as representative of all other periods. The algorithm run one time for each different value of P and the number of iterations is always 1000. The risk measure's parameters are fixed at $a = 0.5$ and $p = 2$. In Figure 4.1 the results are shown.

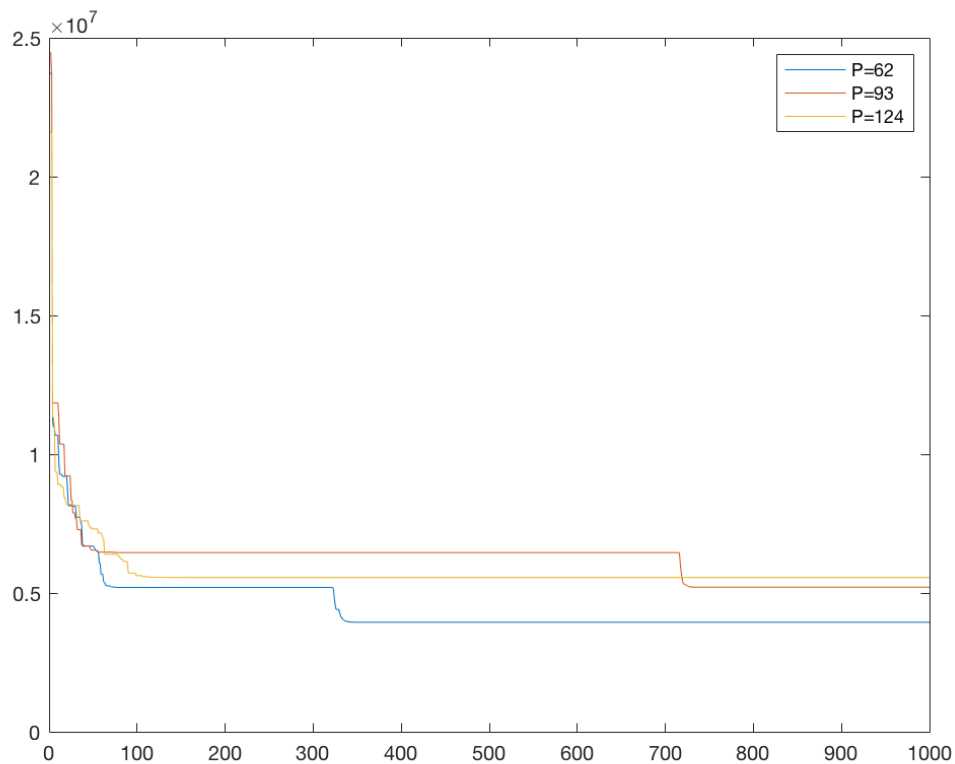


Figure 4.1: PSO with 1000 iterations and different values of the number of particles P .

In this figure, it is possible to see the convergence of the swarms with three different number of particles. It is interesting to note that the best performing swarm is the one with $P = 62$, because it is the one that minimizes the most the fitness function. The second best in this case is the swarm with $P = 93$, followed by the one formed by 124 particles. This graph confirms that choosing a swarm composed by $P = 62$ particles was a good decision. I would like to remember that results of analysis done with metaheuristics always change because they are not an

exact science. This means that in this particular case the best swarm is the one with fewer particles, but if the analysis is made again the results may change.

In order to conclude the analysis of PSO performances, a last consideration is made. I decided at the beginning to use 1000 iterations for every run because it is an intermediate number of iterations. In theory, if the number of iterations is too small, the swarm could be not able to converge very well because it has not sufficient time. A high number of iterations, on the other hand, should provide a better convergence but it is more time consuming, and one of the main advantages of metaheuristics is their capacity to find good solutions in a limited time. The number of iterations equal to 1000 has been chosen also because, as we will see next, a higher number of iterations would have been way too time consuming in the application of GAs. To test the efficiency of 1000 iterations with respect to PSO, the convergence with 1000 iterations is compared with the convergence deriving from two other numbers of iterations: 500 and 2000. In Figure 4.2 it is represented graphically.

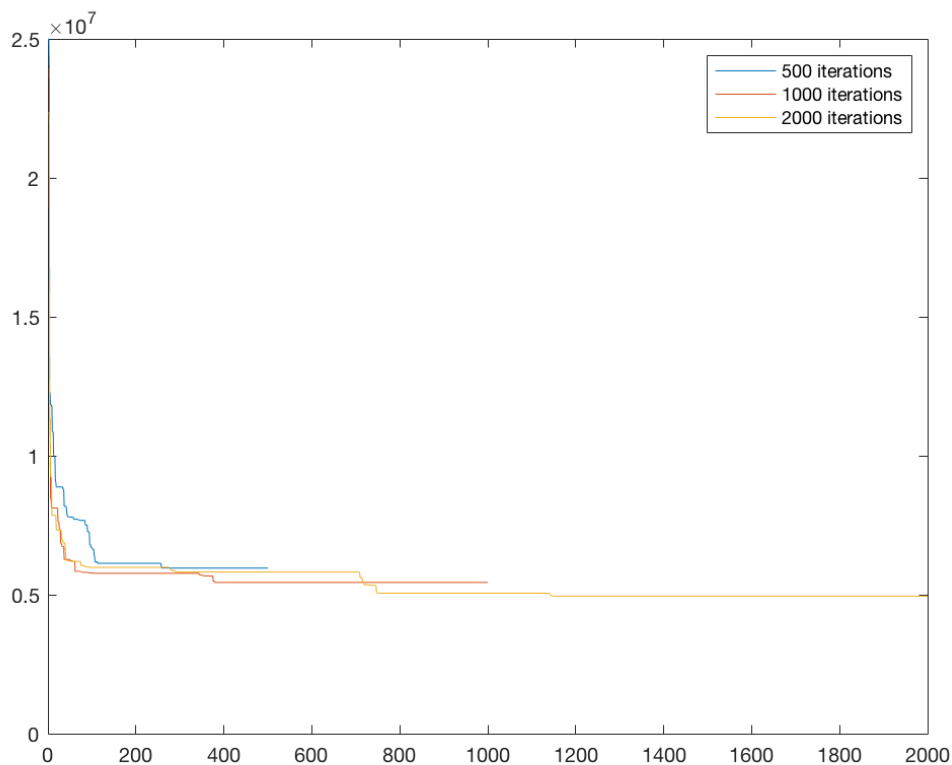


Figure 4.2: PSO with $P = 62$ particles and different numbers of iterations.

As we can see from Figure 4.2, the results are as expected. The better convergence is provided by the higher number of iterations, in this case 2000, while the

worst convergence is the one from the run with only 500 iterations. Looking at them all together, the difference between this three values is not so big, they all provide good convergence. Considering that 2000 iterations will take the double computation time with respect to 1000 iterations and that the results are not significantly different, it is possible to conclude that the choice of 1000 iterations is a good one.

Now that the analysis regarding PSO is concluded, I am going to analyze the portfolio optimization made using GAs. I would like to remember that the values used may not be the optimal values suggested by GAs, but I used them in order to make the performances with PSO and GAs comparable. In particular, the population size suggested by the GAs app in Matlab is 200, but I used 62 like in PSO. Furthermore, I used 1000 iterations while the number of iterations suggested is 100 for every variable, that in this case consists in 6200 iterations since the variable considered are 62 (31 x , the number of securities, and 31 z , the binary variable that tells if the asset is in the portfolio or not).

The analysis made with GAs is going to be different with respect to the one of PSO, since it was not possible to perform the same analysis. As I launched the first run of a GA with 1000 iterations, I immediately found a big drawback of GAs with respect to PSO: computation time. A run with 1000 iterations with PSO takes on average 30 seconds, while a run with the same settings with GAs takes on average 19 hours. Note that 19 hours are 68400 seconds, that means that GAs take more than 3 magnitudes to complete a run with respect to PSO. Due to this huge drawback, it was not possible to perform 10 runs for every value of p , a and K_u . I decided to take into consideration just four portfolios: the two worst portfolio and the two best portfolios provided by PSO. From Table 4.8 it is possible to see that the portfolio with the worst out-of-sample performance, that is -0.148% , is in period 3, with $K_u = 10$ and $p = 1$, and the second worst (out-of-sample performance of -0.147%) is always in period 3 with $K_u = 20$ and $p = 5$. The best performing portfolio, with a 2.068% return, is in period 7 with $K_u = 10$ and $p = 5$. The second best portfolio is provided when $K_u = 20$ and $p = 5$, but I decided not to analyze it, since it has the same value of p . I decided in stead to analyse the portfolio that is the third best, which occurs always in period 7, but it is when $K_u = 10$ and $p = 2$, to see if a less cautious investor is able to reach a higher performance or not. I will rename the portfolio, so that misunderstandings are not created, in order from the worst one to the best one as follows:

- Portfolio 1: period 3, $K_u = 10$, $a = 0.5$, $p = 1$;
- Portfolio 2: period 3, $K_u = 20$, $a = 0.5$, $p = 5$;

- Portfolio 3: period 7, $K_u = 10$, $a = 0.5$, $p = 2$;
- Portfolio 4: period 7, $K_u = 10$, $a = 0.5$, $p = 5$.

One run with 1000 iterations is performed for each of the four portfolios selected. In Table 4.9 the returns of the out-of-sample performances are shown.

Portfolio return (%)	
Portfolio 1	-9.30884%
Portfolio 2	-11.79457%
Portfolio 3	22.57522%
Portfolio 4	66.10033%

Table 4.9: Portfolio out-of-sample returns performed with GAs.

It is possible to see immediately that the results are extremely high or extremely low compared to the results obtained with PSO, and that these values are almost impossible to reach, specially in three months, in the real world. A positive consideration is that the worst portfolios still provides losses in the investment period, and the best portfolios still have (irrealistic) positive returns. With respect to the value of p , in the case of portfolio 3 and 4, it is possible to say that a more cautious investor is going to have a higher return, but considering portfolio 1 and 2, it is possible to say the opposite, a risk-seeker investor is going to lose less with respect to a careful investor. Taking this into account, it is not possible to make meaningful considerations about the perception of risk of the investor and the performances of the portfolio.

Let's try to understand why the application with GAs leads to such results. First of all consider that just one run have been performed for each portfolio, and that being GAs a metaheuristic, this result can be particularly good, bad or in the middle. This, though, does not explain the unrealistic results. In order to find an explanation I will look at the constraint, in particular if they are violated or not. The first constraint in the portfolio is the return constraint, that is largely violated by portfolio 1 and 2.

As we can see form Table 4.10, portfolio 1, 3 and 4 strongly violate the upper bound of the cardinality constraint. For these portfolios the maximum number of assets allowed in the portfolio is 10, but they are composed respectively by 18, 16 and 16 assets. As seen previoulusly, the violation of this constraint is not that worrying, as long as the other constraint are not significantly violated. In Table

	Number of assets	$\rho_{a,p}$
Portfolio 1	18	0
Portfolio 2	18	0
Portfolio 3	16	0
Portfolio 4	16	0

Table 4.10: Number of assets and risk measure for each portfolio performed with GAs.

4.10 it is also possible to see the value of the risk measure. The risk of all four portfolios is equal to zero, this is another unrealistic result. The objective of the portfolio is to minimize the risks so in theory the algorithm performed well, but in the real world it is not possible to find a portfolio composed by risky assets that is riskless. In order to discuss the other constraints, let's have a look at the compositions of the four portfolios.

In Table 4.11 are reported, for each of the four portfolios, the percentage weight of capital invested in each asset and the values of the binary variable z . It is possible to see immediately that these values are not consistent with the constraints of the portfolio, and are highly unrealistic. Firstly, consider the fraction allowed to invest in each asset. As initial condition it has been set that the minimum fraction allowed is $d = 0.02$ and the maximum fraction allowed is $u = 0.2$. Almost all of the fraction invested are not inside these two values or at least in the nearby. On the contrary, the values are extremely high or extremely low. Furthermore, the positive value of d , sets also a prohibition on short-selling, prohibition that is highly violated. For example, the highest weight in portfolio 1 is 9.88795 (invested in asset 24), this means that in that asset it is invested the 988.795% of the capital available. The lower weight is given by asset number 8, and it is -8.30404, which means that in this asset the investor has a short position of 830.404% of its initial budget. It is easily understandable that this makes no sense. Another discrepancy is provided by the variable z . Note that when $z = 1$ the asset should be included in the portfolio and when $z = 0$ the asset should not be included in the portfolio and thus its weight should be zero. In the portfolio created with GAs, there is a weight for every asset. This means that also the last constraint of the portfolio is violated. A last consideration is made regarding the budget constraint. It is no surprise that with the unrealistic values of the percentages of value invested the budget constraint is also violated, as reported in the last row of Table 4.10. The sum of the weights invested in each asset should be one in order to invest the whole initial budget.

Portfolio 1, 2 and 3 invest respectively 55, 39 and 10 times the budget available. Portfolio 4, on the other hand, has a short position of 25 times the initial budget. This is a very severe constraint violation.

	Portfolio 1		Portfolio 2		Portfolio 3		Portfolio 4	
Asset	Weights	z	Weights	z	Weights	z	Weights	z
x_1	9.451632366	0	0.350650642	0	-5.43282212	1	-5.575949494	0
x_2	-7.029498044	1	-5.884034855	1	4.437757153	1	2.965765507	0
x_3	3.429151212	0	2.794965589	0	5.492067521	1	-3.223170495	1
x_4	-0.610420183	0	4.09550897	0	2.328500376	0	-2.476700066	0
x_5	0.58116698	0	8.25142798	0	7.982946516	1	8.630978778	1
x_6	2.800396401	1	-2.929681597	1	4.940845093	0	-2.114384349	1
x_7	3.840284463	0	-0.522207694	1	-0.548476358	0	-9.312625465	1
x_8	-8.304037911	1	6.435331802	1	-6.662376721	0	0.643429264	1
x_9	-7.384832506	1	-0.54077964	1	-4.42035952	1	2.857855864	0
x_{10}	9.286960114	0	4.455790492	0	-1.822545488	1	-3.650102196	1
x_{11}	5.273645779	0	-9.707438264	1	3.788804009	1	-3.557434781	0
x_{12}	6.175884237	1	6.089959157	0	0.609181089	0	9.531352236	1
x_{13}	9.516520167	0	5.311021681	1	-6.42862576	0	-8.093610706	0
x_{14}	-1.283340561	1	1.514509944	1	-4.941935715	1	1.183830787	1
x_{15}	-2.839792064	0	8.792656871	1	-5.562746489	0	-5.564052103	0
x_{16}	-3.124034717	1	1.351871575	1	-8.749173494	1	6.280779091	0
x_{17}	9.187101424	0	-5.260673806	1	-8.679715192	1	-4.990315801	0
x_{18}	5.05490928	1	1.309910275	0	8.457120239	0	-6.876169106	1
x_{19}	0.236778211	1	1.363921031	0	2.69907524	0	-9.483874588	1
x_{20}	8.325174668	1	0.708616535	1	0.049664939	0	-0.901463391	1
x_{21}	-4.364850238	1	-4.94631636	1	2.350232295	0	9.165995415	0
x_{22}	-1.198021096	1	8.360731805	1	6.858459695	1	9.089011349	1
x_{23}	7.9592269	1	8.791655323	1	-0.762734914	1	-6.716081653	0
x_{24}	9.887954256	0	2.171829615	1	1.453619077	1	-0.218985072	0
x_{25}	-3.492092233	1	-9.142834419	0	7.645471256	1	5.170058519	0
x_{26}	5.520685406	1	-1.890533517	0	0.039980102	1	-5.698830754	1
x_{27}	-3.490012654	1	-3.580107749	0	8.765257406	0	-8.017703191	1
x_{28}	-8.236414356	0	8.424145725	1	1.018623262	1	0.232249313	1
x_{29}	1.822289158	1	-6.596592236	0	0.838994822	0	-1.57123163	0
x_{30}	2.010478454	1	0.210267669	1	-4.186888979	0	6.594240046	1
x_{31}	6.461698604	0	9.263263335	0	-1.206448346	0	0.56399462	0
$\sum_{i=1}^{31} x_i$	55.46459		39.04684		10.35175		-25.13314	

Table 4.11: GAs -Percentage of capital invested in each assets and the variable z for each portfolio.

To find an explanation to such results let's look at the convergence graph of

all of the four portfolios. In this graphs, it is represented with black dots the best fitness and with blue dots the mean fitness. In theory, the mean fitness should converge toward the best fitness, and thus the graphs should have a funnel shape.

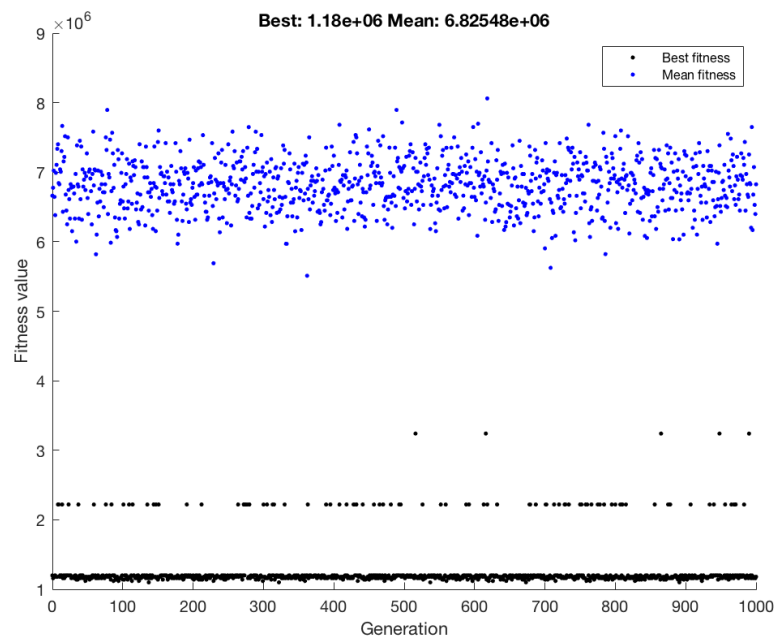


Figure 4.3: Portfolio 1 convergence performed with GAs, with populazion size 62 and 1000 iterations.

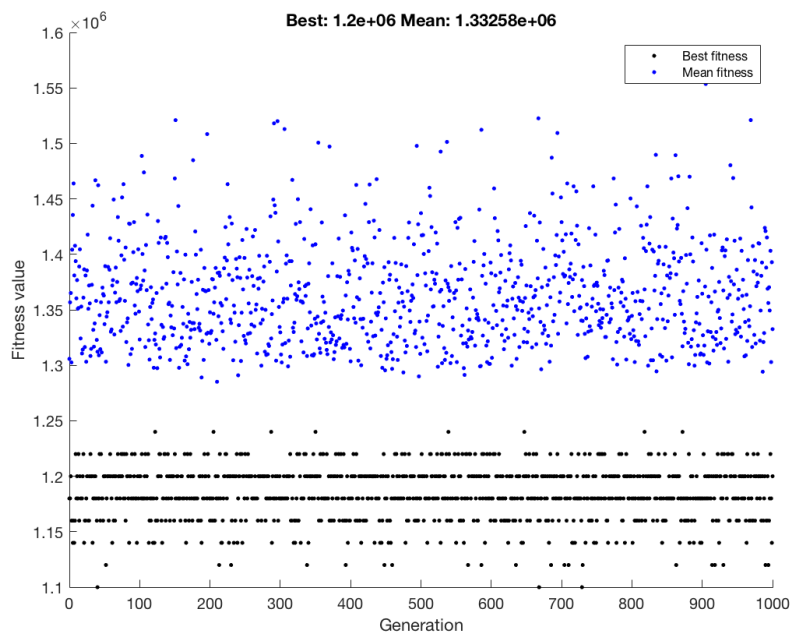


Figure 4.4: Portfolio 2 convergence performed with GAs, with populazion size 62 and 1000 iterations.

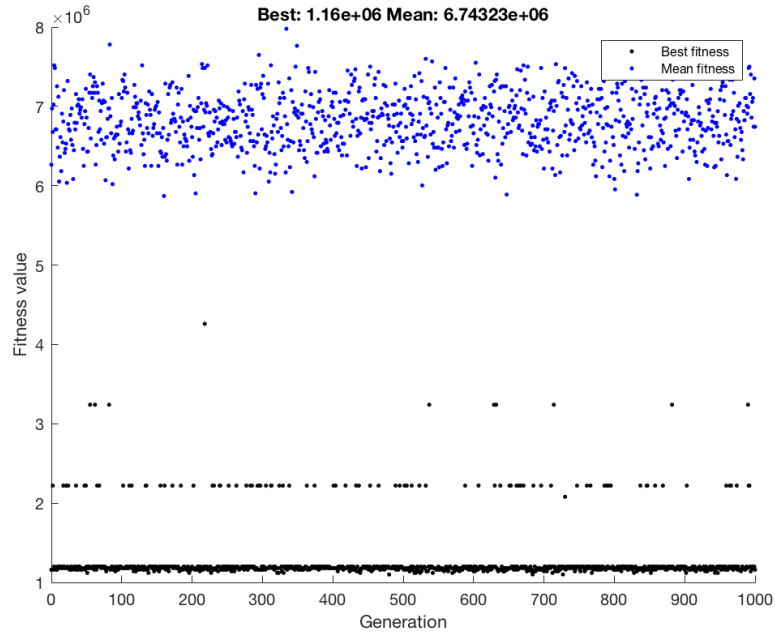


Figure 4.5: Portfolio 3 convergence performed with GAs, with population size 62 and 1000 iterations.

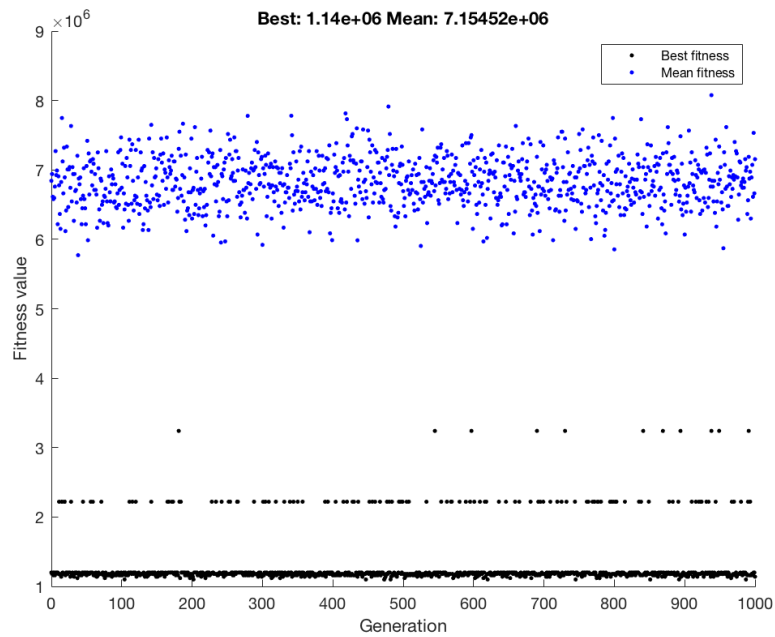


Figure 4.6: Portfolio 4 convergence performed with GAs, with population size 62 and 1000 iterations.

As it is possible to see, in none of the four graphs there is a hint to convergence: best fitness and mean fitness create parallel lines that do not converge. The possible explanation to this lies in the number of iterations. For a genetic algorithm to

converge are necessary a lot more than 1000 iterations, possibly 20000. With this few iterations the algorithms is not able to perform well, it is just a warm up. This is why GAs provided unrealistic results in this analysis, the algorithm was not in optimal conditions to operate. As previously stated, it is suggested to use 100 iterations for each variable and a population size of 200 when the variables are more than 5. In this analysis the variables considered were 62, the population size used 62 and the number of iteration 1000. These values are much lower that the one suggested. It was not possible to prove if with the values suggested GAs perform well, because it would have take more or less 5 days for one run. It is possible to conclude that GAs provide unacceptable portfolios.

Conclusions

In this analysis, two metaheuristic methods have been used for portfolio optimization. For my work, I decided to make an analysis that reflects the real financial world as much as possible. To this intent, the portfolio selection model used is a model that tries to represent reality, taking into consideration everyday problems of the financial world, like transaction costs and taxes. The risk measure used is different from the ones suggested by literature for portfolio optimization problems. The two-sided risk measure adopted has been chosen because it also fits better in the real world. With this measure an investor is able to consider positive and negative risk simultaneously, and it is also possible to give weights to them, in order to represent in an optimal way the risk attitude of each possible investor. Furthermore, it has been decided to use metaheuristics because of their great advantage: the limited computation time. This is very useful since timing in financial markets is crucial.

The results obtained from this analysis are not as good as expected in general. Regarding PSO, the results are quite good and they fit with the real world. The optimal portfolios obtained does not have outstanding performances but this does not depend exclusively on the metaheuristic. One reason for this results are the data taken into consideration. The Italian economy in general and, as a consequence, the markets did not perform particularly well in the time period considered. The PSO algorithm can also be improved in order to provide better results. For sure a higher number of iterations, such as 2000 or 5000, would have lead to better portfolio performances in the investment period, but then it would have been impossible to make a comparison on equal terms between PSO and GAs. The greatest advantage of PSO is its computation time. The solutions obtained are good and the time needed to get them is very low. This is how I expected since the theory tells that metaheuristics are methods that provide good solutions in a limited time.

With respect to GAs, on the other hand, the results obtained are not even close to reality, it is not possible to obtain such portfolios in the real world. GAs minimized the risk so much that it has become null. By doing this they violated all the constraints of the portfolio in a severe way, and the resulting out-of-sample

performances are not credible. As a consequence, the portfolios obtained with GAs are not eligible. Of course GAs do not provide always results such these. For our case there is a wide room for improvement, by setting a bigger population size and a higher number of iterations the solutions provided by GAs could have been better. Here the main drawback of GAs arises. With low values of population size and iterations a run takes 19 hours on average, with higher values the time will increase. When using metaheuristics an investor is willing to accept a good solution that is not the optimal one, provided that the time needed to have this solution is limited (and less than when exact methods are used). Given the high computation time needed and the fact that the solution is not the best one (even when the algorithm runs with optimal settings), I would recommend an investor not use GAs for portfolio optimization problems.

These considerations prove why there is an always increasing research made with respect to PSO. Specially in recent years, this metaheuristic has been more and more studied and applied in different fields. This is due to the fact that it is fast, easy to implement and the solutions that it provides are good. Regarding GAs, the computations underneath them are more complex and thus more time consuming. At the beginning I thought that this could have been an advantage, because it could have lead to more accurate solutions, even if the time required was more. After performing the analysis I understood that the time requested was much higher than expected and that compared to the time needed by PSO, it was way too much. Furthermore, the solution obtained proved that more complex computations are not always synonymous of better results. In fact, the portfolios obtained with GAs are so unrealistic that they can not be compared with PSO's portfolios.

To conclude, I can say that for a portfolio optimization problem I would suggest the use of PSO, but advise against the use of GAs.

Acknowledgements

I would like to thank everyone who helped me complete this dissertation.

First of all, I would like to sincerely thank my supervisor, professor Marco Corazza, whose expert and patient assistance guided and helped me throughout this work. He has been there, available to support me, everytime I had a problem or doubt.

Besides my supervisor, I would like to thank my parents and my sister that lived with me the joys and the labors of studying and for supporting me throughout all these years.

I will always be grateful to my grandfather for having thought about my studies years before, and my grandmother that supported my studies so that I had not to worry about anything else.

Last but not least, I would like to thank Matteo, for the personal and technical support and for being always there for me.

Bibliography

- [1] H. M. Markowitz (1952), "Portfolio Selection", *Journal of Finance*, 7:77-91
- [2] H. M. Markowitz (1991), *Portfolio Selection: efficient diversification of investments*, Basil Blackwell
- [3] H. M. Markowitz (1987), *Mean-Variance Analysis in Portfolio Choice and Capital Markets*, Basil Blackwell
- [4] D. G. Luenberger (1998), *Investment Science*, Oxford University Press
- [5] M. Corazza, D. Favretto (2007), "On the existence of solutions to the quadratic mixed-integer mean-variance portfolio selection problem", *European Journal of Operational Research*, 176:1947-1960
- [6] M. Ehrgott, K. Klamroth, C. Schwehm (2004), "An MCDM pproach to portfolio optimization", *European Journal of Operational Research*, 155:752-770
- [7] M. Andramonov, M. Corazza (2002), "Mixed-integer non-linear programming methods for mean-variance portfolio selection", *Rendiconti per gli Studi Economici Quantitativi*, 2001:21-34
- [8] S. Rachev, S. Ortobelli, S. Stoyanov, F.J. Fabozzi, A. Biglova (2008), "Desirable properties of an ideal risk measure in portfolio theory", *International Journal of Theoretical and Applied Finance*, 11:19-54
- [9] A.J. King (1993), "Asymmetric risk measures and tracking models for portfolio optimization under uncertainty", *Annals of Operation Research*, 45:165-177
- [10] G. Szegö (2005), "Measures of risk", *European Journal of Operational Research*, 163:5-19
- [11] P. Artzner, F. Delbaen, J. Eber, D. Heath (1999), "Coherent Measures of Risk", *Mathematical Finance*, 9:203-228

- [12] R.T. Rockafellar, S. Uryasev (2002), "Conditional value-at-risk for general loss distributions", *Journal of Banking and Finance*, 26:1443-1471
- [13] C. Acerbi, D. Tasche (2002), "On the coherence of expected shortfall", *Journal of Banking and Finance*, 26:1487-1503 title = On the coherence of expected shortfall,
- [14] C. Acerbi, D. Tasche (2002), "Expected Shortfall: a natural coherent alternative to value at risk", *Economic Notes by Banca Monte dei Paschi di Siena SpA*, 31:379-388
- [15] Z. Chen, Y. Wang (2008), "Two-sided coherent risk measures and their application in realistic portfolio optimization", *Journal of Banking and Finance*, 32:2667-2673
- [16] M. Corazza, G. Fasano, R. Gusso (2013), "Particle Swarm Optimization with non-smooth penalty reformulation, for a complex portfolio selection problem", *Applied Mathematics and Computation*, 224:611-624
- [17] A. Brabazon, M. O'Neill (2006), *Biologically Inspired Algorithms for Financial Modelling*, Springer
- [18] Xin-She Yang (2010), *Nature-Inspired Metaheuristic Algorithms*, 2nd Edition, Luniver Press
- [19] K. Sörensen, F. Glover (2010), *Advanced Metaheuristics for High-Level Synthesis*, Springer
- [20] G. Di Tollo, A. Roli (2007), "Metaheuristics for the Portfolio Selection Problem", *International Journal of Operations Research*, 5:13-35
- [21] C. Blum, A. Roli (2003), "Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison", *ACM Computing Survey*, 35:268-308
- [22] K. Sörensen, M. Sevaux, F. Glover (2017), *A History of Metaheuristics*, Springer
- [23] R.L. Haupt, S.E. Haupt (2004), *Practical Genetic Algorithms*, 2nd Edition, Wiley
- [24] M. Mitchell (1998), *An Introduction to Genetic Algorithms*, The MIT Press
- [25] K. Sastry, D. Goldberg, G. Kendall (2005), *Genetic Algorithms: a tutorial*, Springer

- [26] L. Altenberg (1995), *The Schema Theorem and Price's Theorem*, Springer
- [27] D. Goldberg, K. Sastry (2001), "A Practical Schema Theorem for Genetic Algorithms Design and Tuning", *Illinois Genetic Algorithms Laboratory*
- [28] J. Kennedy, R. Eberhart (1995), "Particle Swarm Optimization", *Proc. of the IEEE International Conference on Neural Networks*, 1942-1948
- [29] R. Poli, J. Kennedy, T. Blackwell (2007), *Particle Swarm Optimization*
- [30] J. del Valle, G. K. Venayagamoorthy, S. Mohagheghi, R. G. Harley (2008), "Particle Swarm Optimization: Basic Concepts, Variants and Applications in Power Systems", *IEEE Transactions on Evolutionary Computation*, 12:171-195
- [31] J. Kennedy, R. Mendes (2003), "Neighborhood Topologies in Fully-Informed and Best-Of-Neighborhood Particle Swarm", *Proc. IEEE Int. Workshop Soft Computing in Industrial Applications*, 45-50
- [32] Y. Shi, R. Eberhart (1998), "A Modified Particle Swarm Optimizer", *Proc. of the IEEE Congress on Evolutionary Computation*, 69-73
- [33] M. Clerc, J. Kennedy (2002), "The Particle Swarm - Explosion, Stability, and Convergence in a Multidimensional Complex Space", *IEEE Transactions on Evolutionary Computation*, 6:58-73
- [34] E. Bonabeau, M. Dorigo, G. Theraulaz (1999), *Swarm Intelligence: From Natural to Artificial Systems*, Oxford University Press
- [35] R. Eberhart, Y. Shi (1998), *Comparison between Genetic Algorithms and Particle Swarm Optimization*, Springer
- [36] Wikipedia, "Newton's laws of motion", <https://en.wikipedia.org/wiki/Newton27s-laws-of-motion>
- [37] T. Chang, N. Meade, J. Beasley, Y. Sharaiha (2000), "Heuristics for cardinality constrained portfolio optimisation", *Computer and Operations Research*, 27:1271-1302
- [38] D. Luenberger, Y. Ye (2016), *Linear and Nonlinear Programming*, Springer