UNIVERSIDADE ESTADUAL DE CAMPINAS
Faculdade de Engenharia Elétrica e de Computação

Pedro Mariano Sousa Bezerra

Multi-objective Optimization based on a Multi-criteria Estimation of Distribution

Otimização multiobjetivo com estimação de distribuição guiada por tomada de decisão multicritério

Campinas

2018

UNIVERSIDADE ESTADUAL DE CAMPINAS
Faculdade de Engenharia Elétrica e de Computação

Pedro Mariano Sousa Bezerra

Multi-objective Optimization based on a Multi-criteria Estimation of Distribution

Otimização multiobjetivo com estimação de distribuição guiada por tomada de decisão multicritério

Dissertation presented to the School of Electrical and Computer Engineering of the University of Campinas in partial fulfillment of the requirements for the degree of Master in Electrical Engineering, in the area of Computer Engineering.

Dissertação apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas como parte dos requisitos exigidos para a obtenção do título de Mestre em Engenharia Elétrica, na Área de Engenharia de Computação.

Orientador (Supervisor): Prof. Dr. Fernando José Von Zuben

Co-orientador (Co-Supervisor): Prof. Dr. Guilherme Palermo Coelho

Este exemplar corresponde à versão final da dissertação defendida pelo aluno Pedro Mariano Sousa Bezerra, orientada pelo Prof. Dr. Fernando José Von Zuben e co-orientada pelo Prof. Dr. Guilherme Palermo Coelho.

Campinas

2018

Agência(s) de fomento e nº(s) de processo(s): FAPESP, 2016/21031-0; CAPES
ORCID: https://orcid.org/0000-0002-1491-145X

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca da Área de Engenharia e Arquitetura
Rose Meire da Silva - CRB 8/5974

So85m

Sousa Bezerra, Pedro Mariano, 1990-
Multi-objective optimization based on a multi-criteria estimation of distribution / Pedro Mariano Sousa Bezerra. – Campinas, SP : [s.n.], 2018.

Orientador: Fernando José Von Zuben.
Coorientador: Guilherme Palermo Coelho.
Dissertação (mestrado) – Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Otimização multiobjetivo. 2. Tomada de decisões. 3. Algoritmos genéticos. 4. Meta-heurística. 5. Misturas - Métodos estatísticos. I. Von Zuben, Fernando José, 1968-. II. Coelho, Guilherme Palermo, 1980-. III. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. IV. Título.

Informações para Biblioteca Digital

Título em outro idioma: Otimização multiobjetivo com estimação de distribuição guiada por tomada de decisão multicritério
Palavras-chave em inglês:
Multiobjective optimization
Decision making
Genetic algorithms
Metaheuristic
Mixture models
Área de concentração: Engenharia de Computação
Titulação: Mestre em Engenharia Elétrica
Banca examinadora:
Fernando José Von Zuben [Orientador]
Renato Antonio Krohling
Levy Boccato
Data de defesa: 23-11-2018
Programa de Pós-Graduação: Engenharia Elétrica

COMISSÃO JULGADORA - DISSERTAÇÃO DE MESTRADO

Candidato: Pedro Mariano Sousa Bezerra RA: 118383

Data da Defesa: 23 de novembro de 2018

Título da Tese: "Multi-objective Optimization based on a Multi-criteria Estimation of Distribution (*Otimização multiobjetivo com estimação de distribuição guiada por tomada de decisão multicritério*)"

Prof. Dr. Fernando José Von Zuben (Presidente, FEEC/UNICAMP)

Prof. Dr. Renato Antonio Krohling (CT/UFES)

Prof. Dr. Levy Boccato (FEEC/UNICAMP)

A ata de defesa, com as respectivas assinaturas dos membros da Comissão Julgadora, encontra-se no SIGA (Sistema de Fluxo de Dissertação/Tese) e na Secretaria de PósGraduação da Faculdade de Engenharia Elétrica e de Computação.

Acknowledgements

I would like to express my sincere gratitude to my advisors, Prof. Fernando José Von Zuben and Prof. Guilherme Palermo Coelho, for the guidance, the patience and the valuable contributions during the process of becoming a master.

I would like to thank as well my parents for the continuous and unconditional support, and my colleagues of the Laboratory of Bioinformatics and Bioinspired Computing (LBiC) for the academic and non-academic discussions.

Finally, I would like to thank the Brazilian funding agencies CAPES and FAPESP (São Paulo Research Foundation, grant #2016/21031-0), for the scholarship that supported the development of this research.

"There are in fact two things, science and opinion; the former begets knowledge, the latter ignorance."

(Hippocrates)

Resumo

Considerando as meta-heurísticas estado-da-arte para otimização multiobjetivo (MOO, do inglês *Multi-Objective Optimization*), como NSGA-II, NSGA-III, SPEA2 e SMS-EMOA, apenas um critério de preferência por vez é levado em conta para classificar soluções ao longo do processo de busca. Neste trabalho, alguns dos critérios de seleção adotados por esses algoritmos estado-da-arte, incluindo classe de não-dominância e contribuição para a métrica de hipervolume, são utilizados em conjunto por uma técnica de tomada de decisão multicritério (MCDM, do inglês *Multi-Criteria Decision Making*), mais especificamente o algoritmo TOPSIS (*Technique for Order of Preference by Similarity to Ideal Solution*), responsável por ordenar todas as soluções candidatas. O algoritmo TOPSIS permite o uso de abordagens baseadas em múltiplas preferências, ao invés de apenas uma como na maioria das técnicas híbridas de MOO e MCDM. Cada preferência é tratada como um critério com uma importância relativa determinada pelo tomador de decisão. Novas soluções candidatas são então amostradas por meio de um modelo de distribuição, neste caso uma mistura de gaussianas, obtido a partir da lista ordenada de soluções candidatas produzida pelo TOPSIS. Essencialmente, um operador de roleta é utilizado para selecionar um par de soluções candidatas de acordo com o seu mérito relativo, adequadamente determinado pelo TOPSIS, e então um novo par de soluções candidatas é gerado a partir de perturbações gaussianas centradas nas correspondentes soluções candidatas escolhidas. O desvio padrão das funções gaussianas é definido em função da distância das soluções no espaço de decisão. Também foram utilizados operadores para auxiliar a busca a atingir regiões potencialmente promissoras do espaço de busca que ainda não foram mapeadas pelo modelo de distribuição. Embora houvesse outras opções, optou-se por seguir a estrutura do algoritmo NSGA-II, também adotada no algoritmo NSGA-III, como base para o método aqui proposto, denominado MOMCEDA (*Multi-Objective Multi-Criteria Estimation of Distribution Algorithm*). Assim, os aspectos distintos da proposta, quando comparada com o NSGA-II e o NSGA-III, são a forma como a população de soluções candidatas é ordenada e a estratégia adotada para gerar novos indivíduos. Os resultados nos problemas de teste ZDT mostram claramente que nosso método é superior aos algoritmos NSGA- II e NSGA-III, e é competitivo com outras meta-heurísticas bem estabelecidas na literatura de otimização multiobjetivo, levando em conta as métricas de convergência, hipervolume e a medida IGD.

Palavras-chaves: otimização multiobjetivo; tomada de decisão multicritério; estimação de distribuição.

Abstract

Considering the state-of-the-art meta-heuristics for multi-objective optimization (MOO), such as NSGA-II, NSGA-III, SPEA2 and SMS-EMOA, only one preference criterion at a time is considered to properly rank candidate solutions along the search process. Here, some of the preference criteria adopted by those state-of-the-art algorithms, including non-dominance level and contribution to the hypervolume, are taken together as inputs to a multi-criteria decision making (MCDM) strategy, more specifically the Technique for Order of Preference by Similarity to Ideal Solution (TOPSIS), responsible for sorting all candidate solutions. The TOPSIS algorithm allows the use of multiple preference based approaches, rather than focusing on a particular one like in most hybrid algorithms composed of MOO and MCDM techniques. Here, each preference is treated as a criterion with a relative relevance to the decision maker (DM). New candidate solutions are then generated using a distribution model, in our case a Gaussian mixture model, derived from the sorted list of candidate solutions produced by TOPSIS. Essentially, a roulette wheel is used to choose a pair of the current candidate solutions according to the relative quality, suitably determined by TOPSIS, and after that a new pair of candidate solutions is generated as Gaussian perturbations centered at the corresponding parent solutions. The standard deviation of the Gaussian functions is defined in terms of the parents distance in the decision space. We also adopt refreshing operators, aiming at reaching potentially promising regions of the search space not yet mapped by the distribution model. Though other choices could have been made, we decided to follow the structural conception of the NSGA-II algorithm, also adopted in the NSGA-III algorithm, as basis for our proposal, denoted by MOMCEDA (Multi-Objective Multi-Criteria Estimation of Distribution Algorithm). Therefore, the distinctive aspects, when compared to NSGA-II and NSGA-III, are the way the current population of candidate solutions is ranked and the strategy adopted to generate new individuals. The results on ZDT benchmarks show that our method is clearly superior to NSGA-II and NSGA-III, and is competitive with other well-established meta-heuristics for multi-objective optimization from the literature, considering convergence to the Pareto front, hypervolume and IGD as performance metrics.

Keywords: multiobjective optimization; multicriteria decision making; estimation of distribution.

List of Figures

List of Tables

Nomenclature

ASF Achievement Scalarizing Function

DM Decision Maker

EDA Estimation of Distribution Algorithm

EMOA Evolutionary Multi-Objective Algorithm

GA Genetic Algorithm

IGD Inverse Generational Distance

MCDM Multi-Criteria Decision Making

MOMCEDA Multi-Objective Multi-Criteria Estimation of Distribution Algorithm

MOO Multi-Objective Optimization

NSGA Non-dominated Sorting Genetic Algorithm

pdf Probability Density Function

ROI Region of Interest

SBX Simulated Binary Crossover

SMS-EMOA $\mathscr{S}$-Metric Selection Evolutionary Multi-objective Optimization Algorithm

TOPSIS Technique for Order of Preference by Similarity to Ideal Solution

Contents

1 Introduction

In the past decades, it has been observed an increase in the demand for finding high quality solutions to problems involving the optimization of multiple objectives, eventually under multiple constraints. These problems arise in relevant research and application areas such as economics (MODIRI-DELSHAD; RAHIM, 2016), logistics (YANG *et al.*, 2015) and engineering (YAO *et al.*, 2016); therefore, more elaborate mathematical formulations are needed to address multiobjective optimization and decision making problems, in order to describe practical optimization problems with high dimension and complexity.

Alongside the development of these formulations, several studies have been conducted aiming at finding solutions for the corresponding mathematical programming problems, covering two well-established and distinct research fronts:

- Multi-Objective Optimization (MOO) (BRANKE *et al.*, 2008), whose goal is to sample the Pareto front in the objective space, composed of solutions presenting a trade-off between their different objective values, allowing the decision maker preferences to be taken into account *a posteriori*, i.e., after the process of finding the solutions.
- Multi-Criteria Decision Making (MCDM) (KÖKSALAN *et al.*, 2011), where the preferences of the decision maker are considered *a priori*, in order to rank multiple alternatives.

In single objective optimization problems, finding the best alternative among candidate solutions is a quite simple process: it suffices to evaluate the objective function. Therefore, there is a global optimal solution and the relative merit of solutions can be directly established. However, when dealing with multiple conflicting objectives, sorting the solutions can be a challenging task, because there is a trade-off between their different objective values. In this case, the concept of dominance is widely used to compare candidate solutions (DEB *et al.*, 2002). A solution $\boldsymbol{x^{(1)}}$ is said to dominate a solution $\boldsymbol{x^{(2)}}$ if all the objective values for $\boldsymbol{x^{(1)}}$ are better than or equal to the ones for $\boldsymbol{x^{(2)}}$, and there is at least one objective for which $\boldsymbol{x^{(1)}}$ has a strictly better value than $\boldsymbol{x^{(2)}}$. The solutions at the Pareto front are non-dominated, i.e., there is no solution dominating any of them, and represent the best set of trade-offs between the objective values for a given problem.

Many techniques have been developed to deal with MOO problems. Among them, we can highlight the adapted versions of single objective optimization solvers to the multi-

objective context, e.g., using scalarizing functions (WIERZBICKI, 1982), as well as optimization meta-heuristics (COELLO COELLO *et al.*, 2007). Essentially, there are two main goals to be fulfilled by an MOO meta-heuristic: to sample solutions as close as possible to the Pareto front, and to guarantee that these solutions are diversified, in order to promote a uniform covering of the Pareto front. Usually, performance metrics are evaluated over the set of candidate solutions obtained by two or more methods to compare how efficient they were in achieving those goals.

Aiming at using the least possible amount of computational resources to achieve the solutions at the Pareto front, most population-based meta-heuristics focus on iteratively finding new candidate solutions that are diverse and non-dominated by the already found solutions. Many proposals have been conceived to decide which candidate solutions will be left behind and which ones will be used as starting points to find new promising candidate solutions. In general, non-dominated solutions are the preferred ones to keep in the population and, given a set of non-dominated solutions, their effective contribution to population diversity in the objective space is traditionally adopted as a secondary sorting criterion, together with performance metrics that emphasize particularities of each application. NSGA-II (DEB *et al.*, 2002) and NSGA-III (DEB; JAIN, 2013), SPEA2 (ZITZLER *et al.*, 2001) and SMS-EMOA (BEUME *et al.*, 2007) are examples of widely used meta-heuristics to solve MOO problems. The first three adopt secondary criteria based on diversity, while the last one uses a criterion that measures the contribution to the hypervolume indicator (ZITZLER; THIELE, 1998), a commonly used metric to compare the performance of different algorithms.

When dealing with MCDM problems, we are interested in sorting a set of alternatives according to multiple preferences of a decision maker (DM). This is not a trivial problem when the criteria used to evaluate each alternative are conflicting. The alternatives can be, for instance, the candidate solutions provided by an MOO algorithm. Not all the solutions at the Pareto front can be of interest to the decision maker. The MCDM technique helps to filter the most suited solutions according to the preferences of the decision maker. This is a scenario in which MOO and MCDM techniques can be combined to solve practical optimization problems in an *a posteriori* decision making approach, i.e., when the user preferences are incorporated after the search.

MOO and MCDM techniques can also be combined prior to or during the execution process (*a priori* and interactive decision making approaches, respectively) in order to guide the search towards regions of interest for the decision maker, assuming that the goal is not to sample the entire Pareto front, but only the portions that matter. This way, computational resources are expected to be better employed, since they can focus on the most relevant solutions among the non-dominated ones. Most MOO meta-heuristics involve multiple decisions

along the population-based search to select the most promising population members. Generally, each decision is associated with user preferences, and multi-criteria decision making techniques may help defining a proper ranking for the current individuals in the population. Many hybrid approaches composed of MOO and MCDM approaches have been proposed in the literature (PURSHOUSE *et al.*, 2014), such as methods based on reference points and weighted techniques. Reference points based methods extend the dominance concept to accommodate aspiration levels, i.e., desired values for the objective functions defined by the decision maker. Hence, non-dominated candidate solutions can be classified by considering how close their objective values are to these reference points. On the other hand, weighted techniques usually incorporate preference information into an achievement scalarizing function (ASF), which must be optimized to guide the search. The multi-objective optimization problem now becomes a single objective optimization task, and the candidate solutions are sorted according to the value of the ASF. In most cases of *a priori* decision making approaches, the decision maker must have some knowledge about the nature of the problem to define his/her preferences, to define the aspiration levels in a reference point approach or the weights in a weighted technique, as an example. If the user does not have any idea about what he is looking for, the hybrid technique will not be as useful as approaches that do without such *a priori* information.

Most MOO meta-heuristics sorts their solutions according to only one preference criterion at a time. A distinct aspect of our approach when compared to other hybrid approaches composed of MOO and MCDM approaches is that the adopted MDCM technique, the TOPSIS algorithm (HWANG; YOON, 1981), allows the use of multiple preference based approaches, rather than focusing on a particular one. Each preference is treated as a criterion with a relative relevance to the DM. It is up to him/her to decide which criterion is more important than the other, and express that in the form of a weighting vector responsible for defining the relative relevance of the criteria. This information is used by TOPSIS to properly rank the candidate solutions. Making the reasonable assumption that better ranked candidate solutions are located in more promising regions of the search space, the sorted list of candidate solutions may be explored to conceive a method to generate new potentially high-ranked candidate solutions.

Generating new individuals is an important part of the search process. Traditionally, evolutionary meta-heuristics apply recombination and/or mutation operators in this step. New members are created by recombining information from the already existing ones. Another interesting alternative is to resort to an Estimation of Distribution Algorithm (EDA) (HAUSCHILD; PELIKAN, 2011), a class of stochastic optimization methods employing estimation of distribution techniques to guide the search for solutions. Instead of using traditional

variation operators, new candidate solutions are generated by building and sampling explicit probabilistic models of promising candidate solutions among the already found ones. The idea behind it is that promising regions of the search space are associated with higher probabilities of generating candidate solutions, which implies that the exploration of the search space will focus on these regions. The use of EDA in optimization problems has proven, in many practical situations, to improve performance both in terms of the demand for computational resources and of the quality of solutions. In this work, we follow this approach and adopt a mixture of Gaussian functions as distribution model, so that each Gaussian distribution is centered at its corresponding current candidate solution and the weights used to compose the mixture are non-negative and add up to unity. The weight of each Gaussian is as high as the relative quality of the corresponding candidate solution, determined by TOPSIS, so that higher quality candidate solutions will contribute more to the probability density function. A roulette wheel operator can then be used to sample a uniformly distributed random number in the interval $[0, 1]$ to select the corresponding Gaussian function, and use it to actually sample the new member. Nevertheless, we adopt a particular sampling scheme, in which new candidate solutions are generated in pairs. Each time, two distinct individuals are selected using the roulette wheel operator, and a new pair of candidate solutions is generated as Gaussian perturbations centered at the corresponding parent solutions. The standard deviation of the Gaussian functions is defined in terms of the parents distance in the decision space, so that the spread of the generated candidate solutions is close to the spread of the parent solutions. This is an interesting property, also observed in a widely used crossover operator in real-coded Genetic Algorithms (GA's), the SBX operator (DEB; AGRAWAL, 1994). Thus, our sampling scheme is able to incorporate the advantages of using a distribution model as well as some of the interesting properties of a powerful operator.

However, the aforementioned assumption may not always be true: sometimes, there may be high quality solutions located in unexplored regions of the search space, that will most likely not be represented by the distribution model. Hence, the process of generating new solutions should be able to eventually reach those regions as well, so that they can be further mapped by the distribution model and thus contribute to the search. This can be achieved, for instance, by resorting to operators that insert randomness in the process, such as a mutation operator.

Essentially, we propose here a hybrid technique denoted by MOMCEDA (Multi-Objective Multi-Criteria Estimation of Distribution Algorithm), combining NSGA-II, NSGA-III, TOPSIS and a mixture of Gaussian functions. The main structure of the evolutionary meta-heuristic follows the one present in NSGA-II and NSGA-III, but endowed with distinct sorting and sampling mechanisms. Sorting is performed by TOPSIS, based on user

preferences derived from state-of-the-art decision policies already proposed as part of well-established meta-heuristics. Sampling is implemented by a mixture of Gaussians, so that regions in the search space containing higher quality candidate solutions tend to be explored more intensively. A special sampling scheme and refreshing operators are also adopted, to help the algorithm reach unexplored promising regions of the search space. Hence, the main goal of this research was to contribute to the insertion of MCDM techniques to promote iterative decision making in MOO population based meta-heuristics, improving its computational performance toward better solutions.

The first chapters of this dissertation focus on presenting the three research areas explored in this work: Multi-Objective Optimization, Multi-Criteria Decision Making and Estimation of Distribution Algorithms. In Chapter 2, we introduce single and multi-objective optimization problems. Chapter 3 is devoted to introduce Multi-Criteria Decision Making problems, emphasizing its applications to support iterative decision making processes in EMOA (Evolutionary Multi-Objective Algorithm) to solve MOO problems. In Chapter 4, we discuss Estimation of Distribution Algorithms (EDA's), a class of evolutionary optimization methods. The main contributions of this research are presented in Chapters 5 and 6. In chapter 5, MOMCEDA is formally introduced. Chapter 6 summarizes the computational experiments performed to evaluate MOMCEDA's performance. The final chapter draws the main conclusions and suggests some ideas for future works.

2 Multi-Objective Optimization

Optimization problems have applications in many disciplines, such as physics, biology, engineering, economics and business. An optimization strategy can be defined as the act of selecting the best option, with regard to some criteria, from a set of available alternatives under given constraints. For instance, finding the best route between two places, considering the limitations imposed by the traffic and the city roads, is an optimization problem. Another example is the problem of finding the dimensions of a building maximizing its area of occupation, given a certain amount of materials.

Mathematical optimization problems are usually characterized by three fundamental elements (WRIGHT, 2016). The first one is a numerical quantity, defined as an objective function, used to evaluate candidate solutions. In many optimization problems, there is only one objective function, and finding the best candidate solution is equivalent to finding the element with the best possible value of the objective function, denoted as optimal solution. The objective function represents a value we wish to minimize (or maximize, depending on the context), such as company's production costs or profits, the duration of an event or the distance between two points of interest. However, there are problems with multiple objective functions that cannot be optimized independently. This kind of problems will be formalized later in this chapter. The second element is a collection of decision variables, which are quantities that can be manipulated in order to optimize the objective functions. As examples, they can be represented by quantities of stock to be bought or sold, the dimensions of a building or the route to be followed by a vehicle through a traffic network. The third element are the constraints, which are conditions that must always be true no matter what the solution is, imposing restrictions on the values that the decision variables can take. For instance, a manufacturing process cannot require more resources than what is available.

The mathematical formulation of an optimization problem is the first step in its resolution. It corresponds to the translation of the problem into equations and inequations that represent its three elements: the objective functions, the decision variables and the constraints. The methods that will be applied to solve the optimization problem will depend on its mathematical formulation. An incomplete or incorrect formulation will lead the method to produce wrong solutions, or no solution at all.

This chapter formally introduces Single and Multi-Objective Optimization (MOO) problems and presents some commonly used methods to address the latter, with an emphasis on evolutionary meta-heuristics.

2.1 Single Objective Optimization

In single objective optimization problems, there is only one objective function, which means there is only one value we want to minimize or maximize. The minimization problem is formulated as follows:

$$\begin{aligned} \min \quad & f(\boldsymbol{x}) \\ \text{s.t.} \quad & g_j(\boldsymbol{x}) \geq 0 \qquad j = 1, 2, ..., J \\ & h_k(\boldsymbol{x}) = 0 \qquad k = 1, 2, ..., K \\ & \boldsymbol{x} = \begin{bmatrix} x_1 & x_2 & \dots & x_n \end{bmatrix}^T \quad \boldsymbol{x} \in \boldsymbol{\Omega} \end{aligned} \tag{2.1}$$

where $f(.) : \boldsymbol{\Omega} \to \mathbb{R}^+$ is the objective function of the problem, $\boldsymbol{x}$ is the array of n decision variables, $\boldsymbol{\Omega}$ is the search space, $g_j(.) : \boldsymbol{\Omega} \to \mathbb{R}, j = 1, ..., J$ are the inequality constraints and $h_k(.) : \boldsymbol{\Omega} \to \mathbb{R}, k = 1, ..., K$ are the equality constraints. A point in the search space that violates these constraints is an infeasible point. A feasible solution that minimizes the objective function is called optimal solution. A maximizing problem can be converted into a minimizing problem by negating the objective function: $\max(f(\boldsymbol{x})) = -\min(-f(\boldsymbol{x}))$. An inequality constraint of the type $g_j(\boldsymbol{x}) \leq 0$ is equivalent to $-g_j(\boldsymbol{x}) \geq 0$. Hence, all single objective optimization problems can be formulated as presented in Eq. (2.1).

A local minimum $\boldsymbol{x}^*$ is defined as an element for which there exists some $\delta > 0$ such that, for all $\boldsymbol{x} \in \boldsymbol{\Omega}$ such that $\|\boldsymbol{x} - \boldsymbol{x}^*\| \leq \delta$, we have $f(\boldsymbol{x}^*) \leq f(\boldsymbol{x})$. In other words, the values of the objective function are greater than $f(\boldsymbol{x}^*)$ in a sufficiently small neighborhood around $\boldsymbol{x}^*$. If $f(\boldsymbol{x}^*) \leq f(\boldsymbol{x})$ for every $\boldsymbol{x}$ in the feasible set of the search space, then $\boldsymbol{x}^*$ is also a global minimum, which is the optimal solution to the optimization problem. Local and global maxima are defined similarly. Because a global optimal solution is also a local optimal solution, some optimization methods will find it difficult to make the distinction between them, and will often treat a local optimal solution as an actual solution to the problem. In some cases, a local optimal solution is good enough; however, knowing that there is a better solution is an important information. Figure 1 shows an example of a unidimensional continuous function and its respective global and local minima and maxima for $0.1 \leq x \leq 1.1$.

The approach that will be used to solve a single objective optimization problem depends on its characteristics. For instance, if there is a discrete and manageable number of feasible candidate solutions, solving the problem is as simple as evaluating the objective function and sorting the candidate solutions accordingly. Also, problems in which the variables are continuous quantities require a different approach from problems in which the variables are discrete or combinatorial quantities. Solving these kinds of problems is out of scope here.

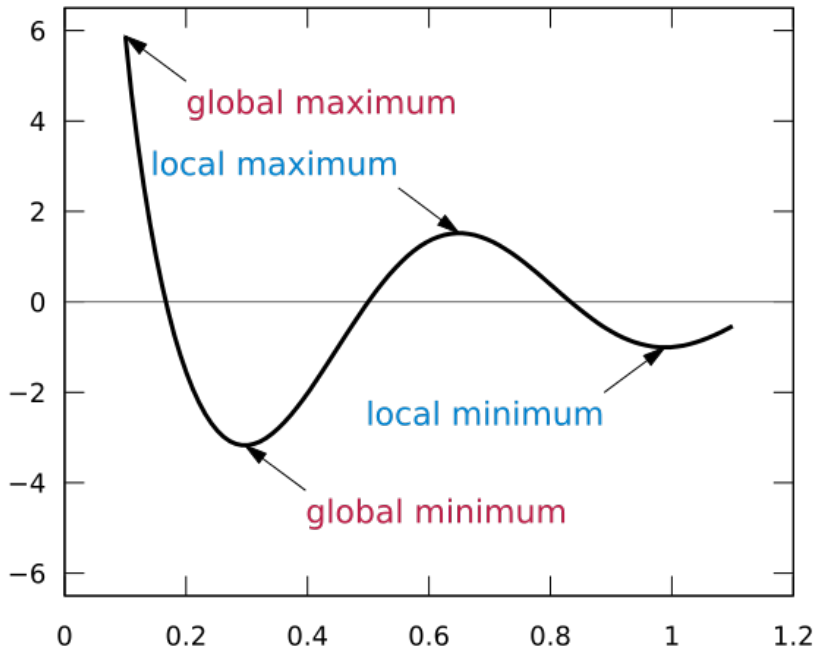

Figure 1 – Local and global maxima and minima for a unidimensional function in the interval $0.1 \leq x \leq 1.1$.

The interested reader can find more about that in RAO (2009).

2.2 Multi-Objective Optimization: Formulation and definitions

In many situations, there is more than one quantity we wish to optimize at the same time and they are not independent optimization problems. For instance, consider the problem of finding the fastest and the shortest route between two points of interest in a city, subject to limitations imposed by the city roads and the traffic flow. Here, there are two quantities to be minimized: the length of the route and the travel time. This is a Multi-Objective Optimization (MOO) problem with two conflicting objective functions. In MOO, the goal is to find solutions minimizing (or maximizing) the values of multiple objective functions, formalized as follows:

$$\begin{aligned} \min \quad & \begin{bmatrix} f_1(\boldsymbol{x}) & f_2(\boldsymbol{x}) & \dots & f_M(\boldsymbol{x}) \end{bmatrix} \\ \text{s.t.} \quad & g_j(\boldsymbol{x}) \geq 0 \qquad j = 1, 2, ..., J \\ & h_k(\boldsymbol{x}) = 0 \qquad k = 1, 2, ..., K \\ & \boldsymbol{x} = \begin{bmatrix} x_1 & x_2 & \dots & x_n \end{bmatrix}^T \quad \boldsymbol{x} \in \boldsymbol{\Omega} \end{aligned} \tag{2.2}$$

where $f_i(.) : \boldsymbol{\Omega} \rightarrow \mathbb{R}^+, i = 1, ..., M$ is the i-th objective function of the problem, $\boldsymbol{x}$ is the array of n decision variables, $\boldsymbol{\Omega}$ is the search space, $g_j(.) : \boldsymbol{\Omega} \rightarrow \mathbb{R}, j = 1, ..., J$ are the inequality constraints and $h_k(.) : \boldsymbol{\Omega} \rightarrow \mathbb{R}, k = 1, ..., K$ are the equality constraints. The feasible set is formed by points in $\boldsymbol{\Omega}$ not violating these constraints. For the same reasons already discussed in the single objective case introduced in Section 2.1, all multi-objective optimization problems can be formulated as presented in Eq. (2.2).

In MOO problems, the objective functions are conflicting, which means that searching for a solution that improves the value of one of the objectives may lead to worse values for at least some other objective. In this case, unlike single objective optimization problems, there is no global optimal solution; instead, there are several solutions that present trade-offs between the values of the different objective functions. Going back to our example at the beginning of this section, the problem can have a solution that minimizes the distance between the two points, but it is not the fastest route because of traffic jam; in this case, there is another route which is faster because it uses less congested streets, but it is longer than the other option. The problem may have a third solution with an even faster or shorter route than the second one, but not both simultaneously, and so on. All these solutions present a trade-off between the values of the two objectives to be minimized: distance and travel time. From the multi-objective optimization perspective, none of the solutions is better than the other. However, any of them is better than a solution with the worst values for travel time and distance, for obvious reasons. In this example, we can realize that comparing candidate solutions in MOO problems is not as simple as in the single objective case. It becomes even more complicated when the number of objective functions gets higher.

Particularly in the case of a reduced number of objective functions, the concept of dominance is of great relevance (DEB *et al.*, 2002). Given two solutions $\boldsymbol{x^{(1)}}$ and $\boldsymbol{x^{(2)}}$, the solution $\boldsymbol{x^{(1)}}$ is said to dominate the solution $\boldsymbol{x^{(2)}}$ if all the objective values for $\boldsymbol{x^{(1)}}$ are better than or equal to the ones for $\boldsymbol{x^{(2)}}$, and there is at least one objective for which $\boldsymbol{x^{(1)}}$ has a strictly better value than $\boldsymbol{x^{(2)}}$. According to the definition presented in Equation (2.2), we say that $\boldsymbol{x^{(1)}}$ dominates $\boldsymbol{x^{(2)}}$ ($\boldsymbol{x^{(1)}} \prec \boldsymbol{x^{(2)}}$) if and only if:

$$\begin{aligned} &i) \quad f_i(\boldsymbol{x^{(1)}}) \leq f_i(\boldsymbol{x^{(2)}}) \quad \forall i \in \{1, ..., M\} \\ &ii) \quad \exists i \in \{1, ..., M\} : f_i(\boldsymbol{x^{(1)}}) < f_i(\boldsymbol{x^{(2)}}) \end{aligned} \tag{2.3}$$

A solution $\boldsymbol{x}$ to a multi-objective optimization problem is said to be efficient, Pareto-optimal or non-dominated if and only if there is no other feasible solution $\boldsymbol{x'}$ for the problem such that $\boldsymbol{x'}$ dominates $\boldsymbol{x}$ ($\boldsymbol{x'} \prec \boldsymbol{x}$). In our route example, the first three aforementioned solutions are non-dominated, while the solution with the worst objective values is dominated by the others. We are interested in finding as many non-dominated solutions as possible,

since they present the best trade-offs available between their objective values.

In MOO problems, there are two search spaces to be considered: the decision space, which contains the decision variables, and the objective space, featuring its corresponding objective values. A point in the decision space has only one corresponding point in the objective space, but many points in the decision space can be mapped to the same point in the objective space. Figure 2 illustrates this idea for bi-dimensional spaces. Note that it is not mandatory for both spaces to have the same dimension. The correspondence between points in the different spaces is represented by the arrows. The feasible set is delimited by the gray area. The optimization search process takes place on the decision space, by manipulating the decision variables, but the relative quality analysis of solutions is carried out in the objective space.

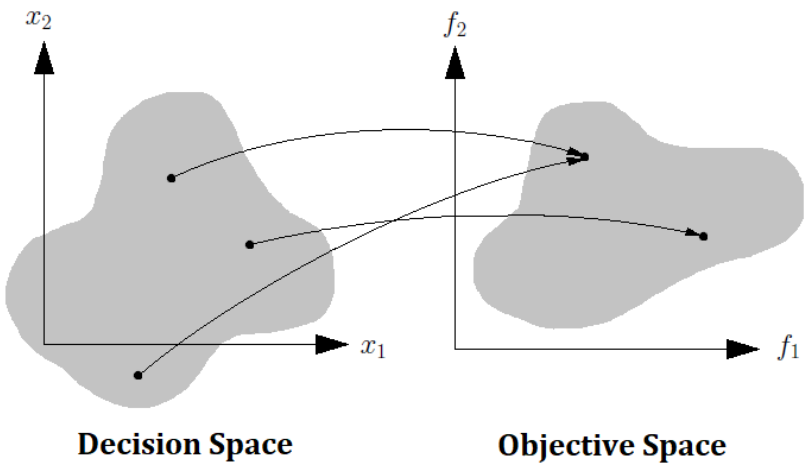

Figure 2 – Example of mapping points between decision and objective spaces.

The set of efficient solutions in the decision space is called *Pareto set*, and its image on the objective space is the *Pareto front*. Formally, the Pareto set is given by:

$$P^* = \{\boldsymbol{x} \in \boldsymbol{\Omega} | \nexists \boldsymbol{x'} \in \boldsymbol{\Omega} : \boldsymbol{x'} \prec \boldsymbol{x}\} \tag{2.4}$$

while the Pareto front is given by:

$$P_F = \{\begin{bmatrix} f_1(\boldsymbol{x}) & f_2(\boldsymbol{x}) & \dots & f_M(\boldsymbol{x}) \end{bmatrix} | \boldsymbol{x} \in P^*\} \tag{2.5}$$

In Figure 3, we see an example of representation of the objective space, considering a minimization problem with two objectives. The Pareto front is represented by a line, with some highlighted efficient solutions represented by triangles, which are non-dominated among

each other. Some dominated solutions, represented by circles, are depicted as well. They are dominated by a subset of solutions at the Pareto front.

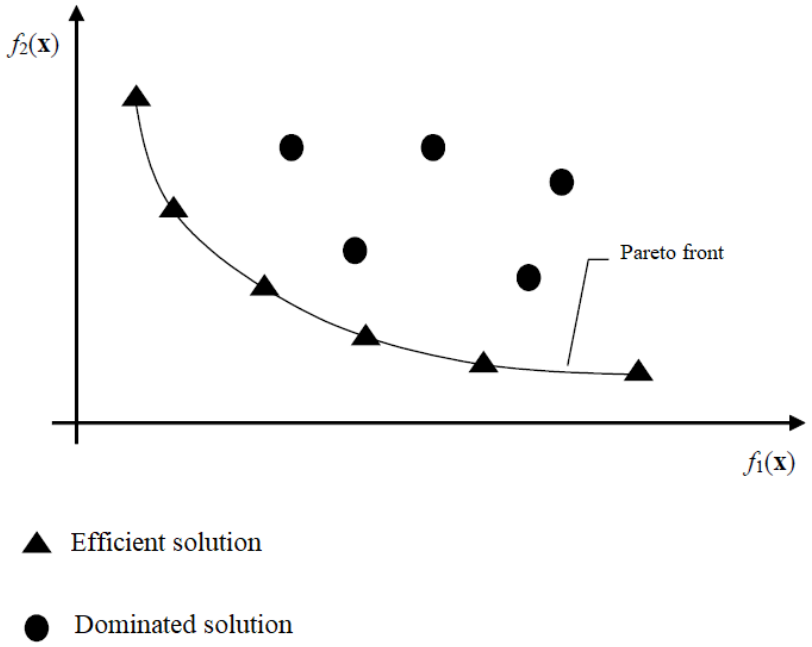

Figure 3 – Illustration of the dominance concept and the Pareto front.

Now that we have established the definitions for relative quality comparison between solutions in MOO problems, we can identify two primary goals for MOO algorithms: to obtain solutions as close as possible to the Pareto front, and to maintain diversity among solutions, such that the Pareto front is uniformly covered. These two goals are illustrated in Figure 4 for a bi-objective minimization problem: the approximation of the Pareto front is represented by the blue arrows, and the coverage is represented by the red arrows.

2.3 Multi-Objective Optimization methods

In order to accomplish the aforementioned goals for MOO problems, a variety of methods have been proposed. They can be roughly divided into two categories: exact resolution methods and meta-heuristics. The methods from the first category make use of mathematical tools (such as gradient and Hessian matrix) to extract information from the decision variables and find solutions with a given precision. Among them, we can highlight the adapted versions of single objective optimization solvers to the multi-objective context, e.g., using scalarizing functions (WIERZBICKI, 1982) to convert the multi-objective optimization problem into a single objective one. Most of the exact methods have poor performance in real-world problems (such as problems with large dimensions, hardly constrained problems, multi-modal

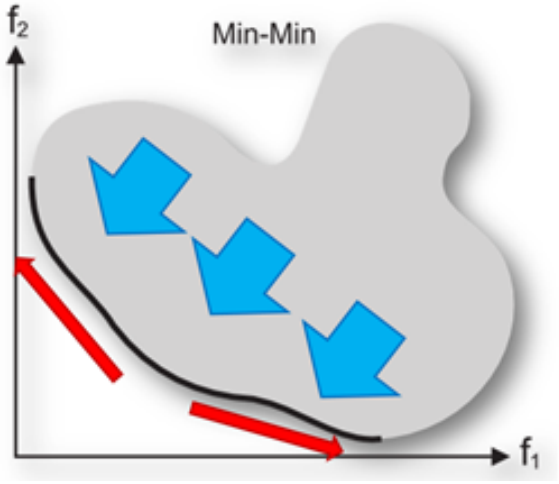

Figure 4 – Illustration of the two primary goals for MOO algorithms (VON ZUBEN; COELHO, 2017).

and/or time-varying problems). On the other hand, meta-heuristic techniques are powerful and flexible search methodologies that have successfully tackled practical challenging problems. Although there is no guarantee that these algorithms will be able to find Pareto-optimal solutions, they are usually able to produce good-quality solutions in reasonable computation times and good enough for practical purposes. In MOO, evolutionary meta-heuristics (COELLO COELLO *et al.*, 2007) are among the most used methods, and represents the main focus of this research.

The idea behind an Evolutionary Multi-objective Optimization Algorithm (EMOA) is to maintain a population of candidate solutions and evolve its members along generations, based on Darwin's Theory of Evolution. At each generation, the population's best members are selected according to some given criteria, and new members are generated from them. The worst members are discarded, and the evolutionary process restarts until it reaches a stopping condition. Aiming at using the least possible amount of computational resources to achieve the solutions at the Pareto front, most population-based meta-heuristics focus on iteratively finding new candidate solutions that are diverse and non-dominated by the already found solutions. Many proposals have been conceived to decide which candidate solutions will be left behind and which ones will be used as starting points to find new promising candidate solutions. In the next sections, we will present some state-of-the-art methods in this category.

2.3.1 NSGA-II

The second version of the *Non-dominated Sorting Genetic Algorithm*, NSGA-II, was proposed in DEB *et al.* (2002) and is one of the most popular MOO methods, due to its

efficiency and good performance in several problems. This population-based evolutionary meta-heuristic exploits the concept of non-dominated fronts. For every generation t of the genetic algorithm, the individuals from the population P_t and the corresponding offspring Q_t, both with the same size N, are sorted according to the dominance relation using a process called *Fast Non-dominated Sorting*. A list $\mathcal{F} = (\mathcal{F}_1, \mathcal{F}_2, ...)$ of non-dominated fronts is then produced. Solutions in a non-dominated front are non-dominated among each other. The first non-dominated front $\mathcal{F}_1$ contains the non-dominated individuals amongst the entire population, being the algorithm's solution for the problem at the end of its execution. The following fronts, if any, are populated by dominated individuals, such that an individual belonging to the front $\mathcal{F}_i, i > 1$, is dominated by at least one individual in the front $\mathcal{F}_{i-1}$. The non-dominated front level of an individual is therefore a criterion to be minimized. Figure 5 illustrates four non-dominated fronts in a bi-objective minimization problem.

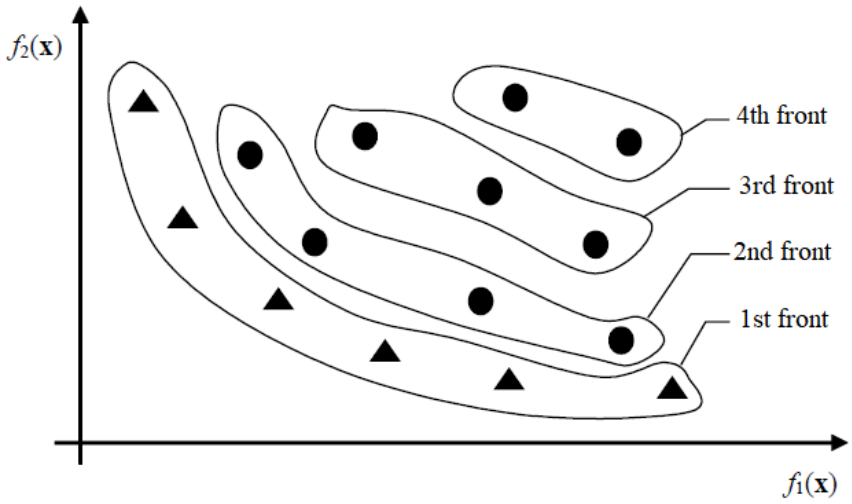

Figure 5 – Non-dominated fronts in a bi-objective minimization problem.

To perform the Non-dominated Sorting, each member of the population is compared with all the others with respect to the dominance relation. The first non-dominated front is formed by the non-dominated members. To find the next front, the members of the previous front are removed from the population, and the next front is formed by the non-dominated members among the remaining ones. The procedure is repeated until all members have been sorted. The *Fast Non-dominated Sorting* procedure proposed in DEB *et al.* (2002) is an optimized version of this algorithm.

After the sorting process, the population of the next generation P_{t+1} is filled with individuals from the first non-dominated fronts, until it is no longer possible to insert all the individuals from a given front k without exceeding the population size. Let $\mathcal{F}_k$ be this front. If P_{t+1} is already complete, the algorithm proceeds directly to the creation of the offspring

population Q_{t+1}. Otherwise, the individuals in $\mathcal{F}_k$ are sorted according to a second criterion, which evaluates the diversity of the population: the crowding distance measure. This measure computes the population's density in the objective space, and attributes higher values to the members located in less populated areas. The remaining vacancies of P_{t+1} are filled by the individuals from $\mathcal{F}_k$ with the highest values of crowding distance, aiming at promoting population diversity.

Once P_{t+1} is complete, the process of sampling new solutions initiates. The parents are selected by binary tournament, whose winner is determined using the same criteria adopted to form the population: non-dominance level and, in case of a tie, crowding distance. The offspring population Q_{t+1} is obtained by applying recombination and mutation operators in the selected individuals. This completes a generation of the algorithm, which will restart the whole process for a new generation unless the stopping condition is achieved. A maximum number of generations or objective functions evaluations are commonly used to stop the execution. The steps performed by NSGA-II are summarized in Algorithm 1. The overall complexity of one generation of the algorithm is $O(MN^2)$, where M is the number of objectives and N is the population size, which is governed by the non-dominated sorting procedure.

Algorithm 1: NSGA-II

```
   Result: Non-dominated individuals of P_t
 1 t = 1;
 2 P_t, Q_t = initialize_population(N);
 3 while stopping condition not satisfied do
 4 |   R_t = P_t ∪ Q_t;
 5 |   F = fast_non_dominated_sort(R_t);
 6 |   P_{t+1} = ∅, i = 1 ;
 7 |   while |P_{t+1}| + |F_i| ≤ N do
 8 |   |   evaluate_crowding_distance (F_i);
 9 |   |   P_{t+1} = P_{t+1} ∪ F_i;
10 |   |   i = i + 1;
11 |   end
12 |   sort_crowding_distance(F_i);
13 |   P_{t+1} = P_{t+1} ∪ F_i[1 : N − |P_{t+1}|];
14 |   Q_{t+1} = offspring(P_{t+1});
15 |   t = t + 1;
16 end
```

2.3.2 NSGA-III

NSGA-III uses a reference-point-based approach focusing on Many-Objective Optimization (MOO problems with more than three objectives) (DEB; JAIN, 2013). Problems with a large number of objectives pose challenges to any optimization algorithm: the proportion of non-dominated solutions in a randomly chosen set of objective vectors becomes exponentially large with an increase in the number of objectives, making it difficult to accommodate an adequate number of new solutions in the population, since non-dominated solutions occupy most of its slots. Also, the diversity-preservation operators, such as the crowding distance operator, are computationally expensive in high dimensions. Instead of searching the entire search space for Pareto-optimal solutions, NSGA-III adopts a predefined multiple targeted search, which helps alleviating the computational burden and the loss of discriminant power associated with non-dominance.

NSGA-III is based on the same concepts adopted by NSGA-II to sort non-dominated individuals, but now a different secondary criterion is used to evaluate diversity within a non-dominance level: the size of an individual's neighborhood, defined as follows. First, the user defines a set of H reference points, or a structured approach is used to create them, with p divisions for each objective axis, in an M-dimensional hyperplane which intercepts each objective axis at 1.0; every reference point gives origin to a reference direction, taking as origin the ideal point of the population in the objective space (the point with the best values for each objective found so far); then, another M-dimensional hyperplane is found by identifying the extreme points in each objective axis, and the objectives are normalized such that the new hyperplane matches the other one marked with the reference points; after that, the individuals are associated with the closest reference direction in the normalized objective space; finally, a neighborhood is formed by the closest individuals to a given direction. Small sized neighborhoods are preferred to maintain diversity. Ideally, one member for each reference point is the desired result at the end of the run. Figure 6 shows an example of $H = 15$ structured reference points distributed over a normalized hyperplane, with $M = 3$ objectives and $p = 4$ divisions. The ideal point and one of the reference directions are depicted as well. The points were created using Denis and Das's systematic approach (DENNIS; DAS, 1998). The total number of reference points (H) in this case is given by:

$$H = \binom{M + p - 1}{p} \tag{2.6}$$

The steps performed by NSGA-III are summarized in Algorithm 2. In line 3, we call the method to build the set Z^s of structured reference points, or we use a set of aspiration levels provided by the user. In the loop from lines 8 to 10, the population S_t is formed by

the first non-dominated fronts of the combined population R_t, and $\mathcal{F}_l$ is the last front that was added to S_t. If P_{t+1} has already N members, then we are done with it; otherwise, the new population P_{t+1} is filled with members from $S_t \backslash \mathcal{F}_l$, and the last K members to be added to P_{t+1} will be chosen from $\mathcal{F}_l$ according to the diversity criterion. In line 18, the *Normalize* procedure performs the objective normalization and produces the normalized objective values $\mathbf{f^n}$ and the set of normalized reference points Z^r. In line 19, the *Associate* procedure associates each population member $\mathbf{s}$ to its closest reference direction $\pi(\mathbf{s})$, where $d(\mathbf{s})$ is the Euclidean distance between $\mathbf{s}$ and $\pi(\mathbf{s})$. In line 20, the neighborhood size ρ for each reference direction is updated by the *niche_count* procedure. Finally, the remaining K members are added to P_{t+1} in line 21 in the *Niching* procedure. The overall worst-case complexity of one generation of NSGA-III is $O(N^2 \log^{M-2} N)$ or $O(N^2 M)$, whichever is larger, where M is the number of objectives and N is the population size.

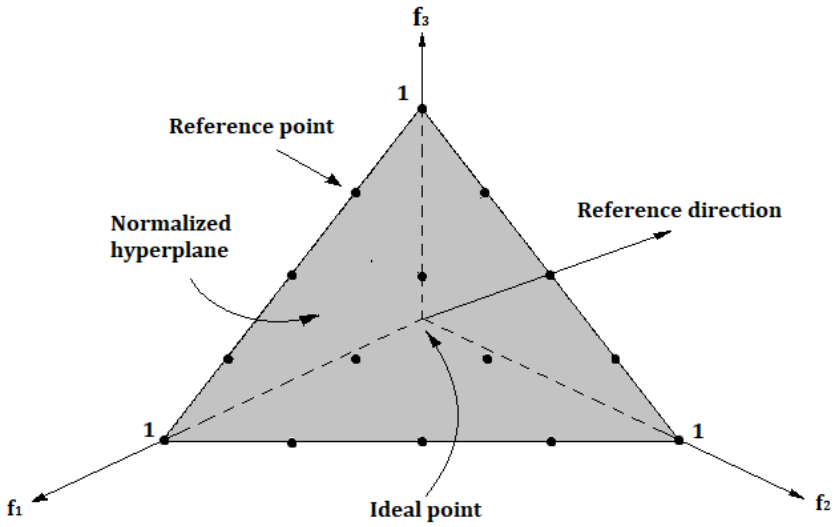

Figure 6 – Distribution of 15 structured reference points in a 3-dimensional normalized hyperplane with 4 divisions for each objective axis.

2.3.3 SMS-EMOA and the hypervolume measure

The *$\mathcal{S}$-Metric Selection Evolutionary Multi-objective Optimization Algorithm* (SMS-EMOA) was proposed in BEUME *et al.* (2007). Again, the concept of non-dominated sorting is adopted to primarily classify individuals, but now combined with a criterion based on the $\mathcal{S}$-metric, commonly known as hypervolume, which is a frequently applied quality measure for comparing the results of EMOA's. This metric rewards the convergence towards the Pareto front as well as the representative distribution of points along the front. The hypervolume

Algorithm 2: NSGA-III

Result: Non-dominated individuals of P_t

$t = 1$;
P_t, Q_t = initialize_population(N);
Z^s = structured_reference_points(M, p);
while *stopping condition not satisfied* **do**
  $R_t = P_t \cup Q_t$;
  $\mathcal{F}$ = fast_non_dominated_sort(R_t);
  $S_t = \emptyset, i = 1$;
  while $|S_t| \geq N$ **do**
    $S_t = S_t \cup \mathcal{F}_i$;
    $i = i + 1$;
  end
  $\mathcal{F}_l = \mathcal{F}_i$;
  if $|S_t| = N$ **then**
    $P_{t+1} = S_t$;
  else
    $P_{t+1} = \cup_{j=1}^{l-1} \mathcal{F}_j$;
    $K = N - |P_{t+1}|$;
    $\mathbf{f^n}, Z^r$ = Normalize(S_t, Z^s);
    $\pi(\mathbf{s}), d(\mathbf{s})$ = Associate(S_t, Z^r);
    ρ = niche_count(Z^r, S_t, F_l, π);
    P_{t+1} = Niching($K, \rho, \pi, d, Z^r, F_l, P_{t+1}$);
  end
  Q_{t+1} = offspring(P_{t+1});
  $t = t + 1$;
end

measure was described as the *size of the dominated space* by its authors (ZITZLER; THIELE, 1998). Let Λ denote the Lebesgue measure (LEBESGUE, 1902), then the $\mathscr{S}$ metric is defined as:

$$\mathscr{S}(B, \mathbf{y}_{ref}) := \Lambda\Big(\bigcup_{\mathbf{y} \in B} \{\mathbf{y}' | \mathbf{y} \prec \mathbf{y}' \prec \mathbf{y}_{ref}\} \Big), B \subseteq \mathbb{R}^M \tag{2.7}$$

where B is a set of points in the objective space and $\mathbf{y}_{ref}$ denotes a reference point that should be dominated by all Pareto-optimal solutions. The exclusive contribution $\Delta_{\mathscr{S}}(s, B)$ of a given point $s \in B$ to the hypervolume measure is defined as:

$$\Delta_{\mathscr{S}}(s, B) = \mathscr{S}(B) - \mathscr{S}(B \backslash \{s\}) \tag{2.8}$$

Given a finite search space and a reference point, maximization of the hypervolume

measure is equivalent to finding the Pareto set. In two dimensions, the hypervolume is equivalent to the dominated area. Figure 7 shows an example of this concept for a set of seven points. In this example, the candidate solution located at the coordinates $(f_1, f_2) = (2, 3)$ has the maximum exclusive contribution to the indicator.

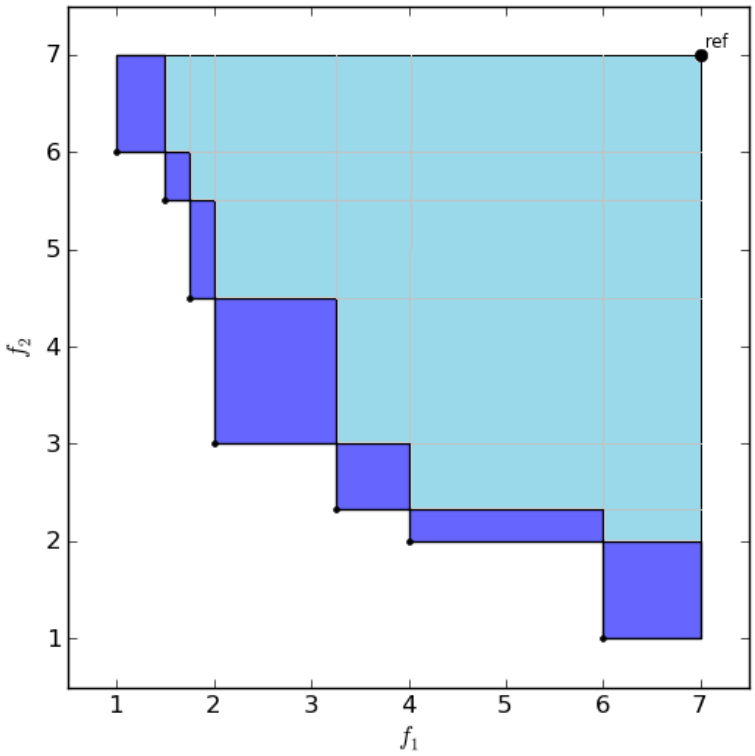

Figure 7 – Bi-dimensional example: the hypervolume is given by the colored area. The exclusive contribution of each point to this metric is highlighted in dark blue.

The steps performed by SMS-EMOA are summarized in Algorithm 3. The algorithm starts with a population P_t of N individuals. A new individual q_{t+1} is generated by means of randomised variation operators. The non-dominated sorting procedure is applied to the population combined with the new individual. Then, the individual from the last front that has the smallest value of exclusive contribution to the hypervolume with respect to that front is removed from the population, and the procedure restarts until it reaches a stopping condition. As expected, the method outperforms most of its concurrents in many MOO problems when the hypervolume is adopted as the performance measure. However, calculating this metric is a computationally expensive operation. The best known algorithms calculate the hypervolume in a runtime that is polynomial in the number of points, but grows exponentially with the number of objectives. This poses limitations to the applicability of the method in

the case of high dimensional problems. For two or three objectives, there are faster algorithms to directly calculate the $\Delta_{\mathscr{S}}(s, B)$ values instead of making repetitive calls of procedures to compute the hypervolume as a whole.

In case of two objectives, we take the points of the worst-ranked non-dominated front $\mathcal{F}_\nu$ and sort them in ascending order according to the values of the first objective function f_1. We get a sequence that is additionally sorted in descending order concerning the f_2 values, because the points are mutually non-dominated. Given a sorted front $\mathcal{F}_\nu = \{s_1, s_2, ..., s_{|\mathcal{F}_\nu|}\}$, $\Delta_{\mathscr{S}}$ is calculated as follows:

$$\Delta_{\mathscr{S}}(s_i, \mathcal{F}_\nu) = (f_1(s_{i+1}) - f_1(s_i)) \cdot (f_2(s_{i-1}) - f_2(s_i)) \tag{2.9}$$

where $i = \{1, ..., |\mathcal{F}_\nu|\}$, $f_1(s_{|\mathcal{F}_\nu|+1}) = f_1(\mathbf{y}_{ref})$, $f_2(s_0) = f_2(\mathbf{y}_{ref})$ and $\mathbf{y}_{ref}$ is the reference point. A fast algorithm for computing all contributions in a three objective space is presented in NAUJOKS *et al.* (2005) and is based on a two-dimensional projection of the set of points. It has a runtime of $O(N^3)$, where N is the population size.

Algorithm 3: SMS-EMOA

Result: Non-dominated individuals of P_t

$t = 1$;
P_t = initialize_population(N);
while *stopping condition not satisfied* **do**
  q_{t+1} = offspring(P_t);
  $R_t = P_t \cup q_{t+1}$;
  $\{\mathcal{F}_1, ..., \mathcal{F}_\nu\}$ = fast_non_dominated_sort(R_t);
  $r = \arg\min_{\mathbf{s} \in \mathcal{F}_\nu}[\Delta_{\mathscr{S}}(s, \mathcal{F}_\nu)]$;
  $P_{t+1} = R_t \backslash \{r\}$;
  $t = t + 1$;
end

2.4 Summary

In this chapter, we introduced single and multi-objective optimization problems. In single objective optimization, we saw that a direct comparison between solutions can be established by evaluating the objective function. In MOO, however, the concept of dominance among solutions is of great relevance. The goal is to find the so-called Pareto front, formed by non-dominated solutions, which represent the best attainable trade-offs between their objective values. We presented some MOO methods, with an emphasis on Multi-objective Optimization Evolutionary Algorithms (EMOA), which are able to produce good quality

solutions in reasonable computation times for practical problems. As we saw, only one criterion at a time is taken into account to classify individuals in these methods: first, the non-dominance level, and in case of a tie, the diversity in the objective space for NSGA-II and NSGA-III, and the exclusive contribution to the hypervolume measure for SMS-EMOA. The three methods presented here were used as basis to elaborate our method. In the next chapter, we will discuss a related problem to the one of finding Pareto-optimal solutions: how to select a final candidate solution according to user preferences. The preference criteria adopted by our method were inspired by the criteria adopted by NSGA-II, NSGA-III and SMS-EMOA.

3 Multi-Criteria Decision Making

Decision making is the process of identifying and selecting an alternative among several possibilities with respect to the preferences of a person, designated as Decision Maker (DM). Decision making is regarded as a cognitive process. In single objective optimization problems, finding the best alternative among candidate solutions is quite simple: we simply evaluate the objective function. Therefore, the relative merit of solutions can be directly established. However, when dealing with multiple conflicting objectives, sorting the solutions can be a challenging task, specially because of the non-domination concept discussed in the previous chapter. Furthermore, the number of Pareto optimal solutions required for describing the entire Pareto optimal front of an MOO problem is usually very large or even infinite. Selecting one preferred solution from all of these solutions is cognitively challenging. Multi-Criteria Decision Making (MDCM) methods can be used to support the decision making steps in MOO methods. In fact, multi-criteria decision making is closely related to multi-objective optimization (DYER *et al.*, 1992).

MDCM methods address complex problems featuring conflicting objectives, where a choice among alternatives is needed. The purpose is to sort the solutions based on the DM preferences. Every decision is made within a decision environment, which is defined as the collection of information, candidate solutions and preferences available at the time the decision must be made. This major class of MCDM methods is further divided into Multi-attribute decision-making, when dealing with discrete candidate solutions, and Multi-objective decision-making, when a theoretically infinite number of continuous candidate solutions is defined (SAN CRISTÓBAL, 2012).

The field of MCDM has developed well-established methods for helping DMs address MOO problems over the last decades (MIETTINEN, 1999). More recently, MCDM tools have been applied together with evolutionary multi-objective optimization algorithms (EMOA) in order to find better solutions for MOO problems. Historically, EMOA tends to emphasize the search for the Pareto optimal set, leaving the task of selecting a single solution as subsequent work for an MCDM method. The MCDM community, however, tends to emphasize the use of preference models either as a precursor to, or during, the search for a single preferred Pareto optimal solution.

This chapter is devoted to discuss MCDM methods as tools to support iterative decision making processes in EMOA to solve MOO problems. In a hybrid technique composed of EMOA and MCDM approaches, the population members are the candidate solutions to be

classified regarding the preference criteria defined by a DM. Multi-attribute decision-making techniques are the most used in these cases, since most EMOAs have a finite number of population members.

3.1 Hybrid composition of EMOA and MCDM techniques

A hybrid composition of EMOA and MCDM methods is characterized by the incorporation of preference information provided by a decision maker, in the search process of an MOO problem. The goal is to explore the search process towards the regions of interest (ROI) for the decision maker, assuming the goal is not to sample the entire Pareto front, but only the portions that matter. Therefore, computational resources are expected to be better employed, since they can focus on the most relevant candidate solutions among the non-dominated ones.

Regarding the moment the decision maker preferences are incorporated, i.e. before, after or during the search, hybrid MOO approaches can be divided into three classes - *a priori*, *a posteriori* and *interactive*, respectively (PURSHOUSE *et al.*, 2014).

In an *a priori* decision making approach, the decision maker expresses his/her preferences before the search process takes place. The provided information is used to replace or supplement the Pareto dominance relation, establishing a total order of the space instead of a partial order as in the non-domination sorting (GOULART, 2014). When the decision maker preferences can be faithfully captured in a mathematical model, an *a priori* method would be effective. However, this is rarely the case. Representative approaches of these methods are based on reference points and weight information, among other forms.

In a reference point based approach, the decision maker preferences are expressed as aspiration levels, representing the desired values for each criterion. The non-domination ranking mechanism is extended to accommodate aspiration levels, and non-dominated solutions can be compared regarding how close they are to the desired values (FONSECA; FLEMING, 1993; MOLINA *et al.*, 2009; DEB; SUNDAR, 2006). A difficulty imposed by this approach is that it requires the decision maker to know the ranges of objective values so as to initialize coherent aspiration levels.

Weights related methods usually make use of an achievement scalarizing function (ASF) to guide the search. The decision maker establishes a relative importance for each criterion. The ASF incorporates the preference information and the candidate solutions can be sorted according to the ASF values. The TOPSIS algorithm (HWANG; YOON, 1981) is one of the most used methods of this class, due to its simplicity and its intuitive rules.

This method will be detailed in the next section. Other examples of this approach include lexicographical ordering, reference direction (DEB; KUMAR, 2007a) and light beam search (DEB; KUMAR, 2007b).

In an *a posteriori* method, the decision maker preferences are incorporated after the search process is completed. In this case, the MOO method first computes an approximation of the Pareto optimal front, and then the MCDM technique finds the preferred solutions according to the DM preferences. This approach may be effective for MOO problems with three objectives or less. However, as the number of objectives increases, this approach becomes inefficient, because solving these problems is computationally expensive and the number of solutions required to properly describe the Pareto front is usually very large in this case. Also, it is cognitively harder for the DM to choose a preferred solution from all these solutions, not to mention the fact that the DM is usually interested only in particular regions of the Pareto front. Among representatives of this class, we can cite modified Pareto dominance relation based EMOAs, such as ϵ-MOEA (DEB *et al.*, 2003), and decomposition based EMOAs, e.g., MOEA/D (ZHANG; LI, 2007).

In an *interactive* decision making approach, the decision maker preferences are incorporated as part of the optimization process. The DM has the opportunity to learn about the problem as the search progresses, and his/her preferences can be refined if necessary. In this case, the DM does not need to have a formidable previous knowledge about the problem like in an *a priori* approach, nor computational resources are wasted processing undesirable solutions like in an *a posteriori* approach. However, the main limitation of this scheme is that the DM may need to be involved intensively during the search process. Representatives of this and other approaches are reviewed in PURSHOUSE *et al.* (2014).

Each hybrid decision making approach has its advantages and disadvantages, and can be efficient or not, depending on many factors: the nature of the problem, the decision maker previous knowledge, the way the preferences are represented, the moment when the preferences are available, among others. In the next section, we will detail the TOPSIS algorithm, the MCDM technique used in our work. This technique is best suited to be used in *a priori* approaches, since it requires the DM to previously establish the relative relevance for each preference criterion, although it could also be used in an *interactive* scheme, allowing the DM to redefine the relative importance of the criteria as the search progresses. It could be difficult, however, to analyze how these changes are influencing the search results.

3.2 TOPSIS algorithm

In our hybrid approach, we want to sort a finite number of multiple candidate solutions to an MOO problem according to the multi-criteria preferences of a decision maker. Among the available tools in the literature, the TOPSIS (*Technique for Order of Preference by Similarity to Ideal Solution*) algorithm (HWANG; YOON, 1981) is one of the most used methods, due to its simplicity and its intuitive rules (BEHZADIAN *et al.*, 2012).

The TOPSIS algorithm sorts multiple candidate solutions by evaluating their distances to ideal solutions in the objective space. The method is based on the concept that the best alternative is considered to be the closest one to the positive ideal solution (which has the best values for all criteria) and as far as possible from the negative ideal solution (which has the worst values for all criteria).

Consider the problem of finding one among Q candidate solutions. Each solution is evaluated considering M criteria. Let $S = \{S_1, S_2, ..., S_Q\}$ be the set of Q candidate solutions and $C = \{C_1, C_2, ..., C_M\}$ be the set of M criteria. The decision matrix is presented as follows:

$$\begin{array}{c} \\ S_1 \\ S_2 \\ \vdots \\ S_Q \end{array} \begin{array}{c} \begin{array}{cccc} C_1 & C_2 & \dots & C_M \end{array} \\ \begin{pmatrix} v_{11} & v_{12} & \dots & v_{1M} \\ v_{21} & v_{22} & \dots & v_{2M} \\ \vdots & \vdots & \ddots & \vdots \\ v_{Q1} & v_{Q2} & \dots & v_{QM} \end{pmatrix} \end{array} \tag{3.1}$$

where v_{ij} represents the evaluation of the i-th candidate solution ($i \in \{1, ..., Q\}$) with respect to the j-th criterion ($j \in \{1, ..., M\}$). The algorithm takes as input the decision matrix and an array $w \in \mathbb{R}^M$ of the criteria relative relevance, provided by the decision maker, where w_j is the numerical value for the relative importance of the j-th criterion. If $w_k > w_j$, it means that the DM considers the k-th criterion to be more important than the j-th criterion. At the end of the execution, the TOPSIS algorithm provides a sorted list of the alternatives (or candidate solutions, in the case of an MOO method). The steps performed by TOPSIS are presented in Algorithm 4.

As an *a priori* decision making approach, the TOPSIS method requires the decision maker to have a sufficient knowledge about the problem in order to establish the relative relevance of the criteria. In some cases, the DM may not know how to properly express the relative relevance in numbers, e.g., the DM may know that the i-th criterion is more important than the k-th criterion, but he/she may not know if it is 2, 5 or 10 times more important.

These values may need to be adjusted, comparing the results of multiple executions. In some situations, however, these values do not need to be extensively tuned.

Algorithm 4: TOPSIS

1 Normalize the decision matrix:

$$r_{ij} = \frac{v_{ij}}{\sqrt{\sum_{k=1}^{Q} v_{kj}^2}} \tag{3.2}$$

2 Calculate the weighted normalized decision matrix:

$$d_{ij} = w_j r_{ij} \tag{3.3}$$

3 Determine the positive ideal solution d_j^+ and the negative ideal solution d_j^- for each one of the M criteria: $d_j^+ = \min(d_{1j}, ..., d_{Qj})$ and $d_j^- = \max(d_{1j}, ..., d_{Qj})$ for minimization, and vice versa for maximization.

4 Calculate the separation distance for each candidate solution:

$$S_i^+ = \sqrt{\sum_{j=1}^{M} (d_j^+ - d_{ij})^2} \quad i = 1, ..., Q \tag{3.4}$$

$$S_i^- = \sqrt{\sum_{j=1}^{M} (d_j^- - d_{ij})^2} \quad i = 1, ..., Q \tag{3.5}$$

5 Calculate the similarity coefficients to the ideal solution for each candidate solution:

$$CC_i = \frac{S_i^-}{S_i^- + S_i^+} \quad i = 1, ..., Q \tag{3.6}$$

6 Sort the candidate solutions in descending order of similarity coefficients: the higher CC_i, the better the i-th candidate solution.

3.3 Summary

In this chapter, we introduced multi-criteria decision making (MCDM) problems, emphasizing its applications to support iterative decision making processes in EMOA to solve MOO problems. Hybrid compositions of EMOA and MCDM techniques are classified according to when the preferences are incorporated: before, after or during the search process - *a priori*, *a posteriori* and *interactive* approaches, respectively. Each one of these classes has advantages and disadvantages. *A priori* approaches can be effective in cases where the number of objectives is low, but they require the decision maker to have a good previous

knowledge of the problem. The other approaches do not have this limitation. *A posteriori* methods, however, waste computational resources with undesirable solutions, specially when the number of objectives is high. This is not an issue in *interactive* approaches, where the decision maker is allowed to learn about the problem as the search progresses and can refine his/her preferences. The problem here is that the DM needs to be intensively involved during the optimization. Finally, we presented the TOPSIS algorithm, which will be used in the method proposed in this dissertation.

4 Estimation of Distribution Algorithms

Over the last few years, the search for solutions to optimization problems in continuous spaces of high dimension has observed a paradigm shift. Not long ago, the state-of-the-art was represented by population-based meta-heuristics with "blind" operators which added a random perturbation to the current solutions, which came along with some local search mechanism to make up for the lack of knowledge about the search space. Three limitations of these meta-heuristics can be highlighted: (1) a high number of parameters to determine, which directly affects the search performance; (2) the need for efficient local search mechanisms; (3) the inability of using the already available knowledge about the search space, obtained from the acquired experience throughout the search.

Lately, some meta-heuristics have chosen to adopt methodologies using estimation of distribution models of the search space, namely EDA (*Estimation of Distribution Algorithm*) (LARRAÑAGA; LOZANO, 2001). The core idea is that promising regions of the search space are associated with higher probabilities of generating candidate solutions, which implies that the search space exploration will focus on these regions. Hence, the search still has a stochastic character, but is oriented by the distribution model, which is derived from the acquired experience throughout the search. The decision making of how to explore the search space is limited to the quality of the already encountered solutions and to the *a priori* knowledge about the nature of the search space, if available.

The practical consequences of this initiative is that EDA's lead to search methods characterized by: (1) a noticeable decrease in the number of evaluation of candidate solutions to reach a given level of performance; (2) an improvement of the search robustness, such that there is not much performance variation through multiple executions of the algorithm for the same problem; (3) the possibility to adjust parameters automatically, derived from the attributes of the distribution model. These properties imply, in many situations, a higher performance both in terms of computational resources and of the quality of solutions. These are the main reasons to support the use of EDA's in multi-objective optimization problems.

On the other hand, building explicit probabilistic models is often more time consuming than using implicit models defined with simple search operators, such as tournament selection and two-point crossover. That is why it may sometimes be advantageous to use implicit models of conventional evolutionary algorithms instead of explicit ones of EDAs. However, doing this is only practical when search operators are available that allow scalable solutions of the target problem class; otherwise, the time complexity of learning a model is a small

price to pay. Furthermore, the discovery of such operators may not be straightforward and it also comes at a cost (HAUSCHILD; PELIKAN, 2011).

This chapter is devoted to introduce Estimation of Distribution Algorithms (EDAs), a class of stochastic optimization methods employing estimation of distribution techniques to guide the search for solutions. EDAs are evolutionary algorithms with a distinct aspect: instead of using traditional variation operators, new candidate solutions are generated by building and sampling explicit probabilistic models of promising candidate solutions, selected from the already found ones.

4.1 Estimation of the probability density function

Consider an application and the data set involved in its process. Extracting information from these data is important to analyze the application's performance. Finding a model that represents the data set helps in the analytical task. Furthermore, new data points can be generated from this model, if desired. One way of building this model is by analyzing the data distribution in the space. The distribution provides important information about spatiality, asymmetry and multi-modality of the density function. The distribution can be modeled with random variables, by estimating the probability density function that best represents the sampled data, employing a given distance metric (GONÇALVES, 2011).

The problem of estimating the probability density function (pdf) of a continuous data distribution requires model identification. These models are essentially divided in parametric and non-parametric distribution models. The former assumes that data distribution follows a known parametric model, and we estimate the model's parameters based on data (e.g., mixture models), while in the latter there is no *a priori* assumption about the data distribution - the density estimation is performed locally with a small number of neighbors (e.g., k-nearest neighbors and kernel-based methods).

There is a variety of methods to determine the estimation of probability density functions. In the class of parametric distribution models, when the data probability distribution is known *a priori*, a simple way of finding the model is by estimating the distribution parameters from its analytical expression. A commonly used method to determine these parameters is the maximum-likelihood estimation given the observations, which can be done depending on the type of the distribution mode. This method is computationally efficient, and despite the fact that data probability distribution is rarely known *a priori*, the method's applicability is not seriously compromised if a sufficiently flexible distribution model is adopted.

Among the non-parametric estimation models, when the data distribution is unknown,

we can highlight the kernel density estimator (PARZEN, 1962; ROSENBLATT, 1956). This method employs kernel functions to estimate the pdf. Commonly used kernel functions are: uniform, triangular, Epanechnikov, normal and cosine. There is a smoothing parameter h called the *bandwidth*, which has a strong influence on the resulting estimate. Restricted Boltzmann machine-based estimation of distribution algorithms are also an example of non-parametric estimation models (SHIM *et al.*, 2013).

4.2 Mixture models

Mixture models belong to the class of parametric distribution models. This implies an *a priori* assumption about the probability density function that represents the data. Finding the model consists of estimating the distribution parameters. A mixture model is formed by a composition of several probability density functions, distinct or not, in order to better represent the data set. This kind of model has been demonstrated as an efficient tool when used in optimization problems (BOSMAN; THIERENS, 2000).

Consider a data set divided in K subsets. Each subset is modeled by a probability density function (pdf), denoted as $f_k, k = 1, ..., K$, and its vector of parameters θ_k is estimated from the data. Depending on the subset importance, a pdf can be more representative than the others. Hence, a weighting coefficient π_k is attributed to each pdf to express how important is the portion of the data that is represented by this function. These coefficients are called mixture coefficients and must satisfy the following relation for the mixture model be a pdf as well:

$$\sum_{k=1}^{N} \pi_k = 1, \quad 0 \leq \pi_k \leq 1 \tag{4.1}$$

where N is the number of functions. When sampling new data from the mixture model, the coefficients indicate the chance that a new data point has been generated by its corresponding pdf.

There are different approaches to estimate the model parameters. As mentioned before, the maximum-likelihood estimation is one of them. The maximization problem can be solved with gradient-based methods or resorting to the well-known and widely used EM algorithm (BISHOP, 2007).

In this research, we propose the use of a Gaussian mixture model to represent the population members of an MOO method. In a Gaussian mixture model, the probability density functions are normal distributions. Here, the model parameters are not estimated,

but adjusted according to the information provided by an MCDM technique, the TOPSIS algorithm described in Section 3.2. Each Gaussian function represents a candidate solution, so each population member is the center of a Gaussian function in the space of decision variables. The standard deviation of each function depends on a pairwise distance to be better explained later. The association between a weight coefficient and the corresponding position of its candidate solution in the ranking follows a predefined function, so that the coefficient is as high as the relevance of its corresponding candidate solution. The principle behind this proposition is that new candidate solutions close to well-ranked current candidate solutions have a higher chance of being generated.

Once the distribution model is built, new candidate solutions can be sampled. Sampling data from a Gaussian mixture model is simple: first, one of the N Gaussian functions is selected by a process similar to a roulette wheel operator: each π_k coefficient corresponds to a portion of the interval $[0, 1]$; it suffices to generate a uniformly distributed random number in this interval to find out which Gaussian function will be used. Then, the chosen Gaussian function is taken to actually sample the new point. In this research, however, we adopted a modified sampling scheme to generate new members: new solutions are sampled in pairs and the Gaussian functions standard deviation is not fixed, but depends on a pairwise distance that will be detailed in the next chapter. This modified sampling scheme has proven to be more efficient in the case of our method.

It is important to note that the whole sampling process takes place in the space of decision variables: we find a Gaussian mixture model to represent the decision variables of each candidate solution, and we use this model to generate new variables representing new candidate solutions. The relative quality of solutions, however, is evaluated in the objective space, as explained in Section 2.2.

Despite the advantages of using EDAs, in some situations the assumption that high quality solutions are located close to well-ranked current candidate solutions may not always be true: sometimes, there may be high quality solutions located in unexplored regions of the search space as well. New points generated by the distribution model will most likely avoid these regions. Furthermore, it is possible for stochastic errors in population sampling to lead to a loss of diversity. If this loss of diversity continues over time (by producing simplified models), it is possible for the population to no longer contain enough information to solve the problem (HAUSCHILD; PELIKAN, 2011). To address this issue, a possible solution is to incorporate operators into EDAs, such as mutation operators (HANDA, 2007). In this work, we apply two refreshing operators to the solutions sampled from the mixture model, aiming at reaching other promising regions of the search space.

4.3 Summary

In this chapter, we introduced Estimation of Distribution Algorithms (EDA's), a class of evolutionary optimization methods where new candidate solutions are generated by building and sampling explicit probabilistic models of promising candidate solutions among the already found ones. Among the advantages of using EDA's, we can highlight an improvement on the performance and robustness of the search for solutions when compared to traditional optimization techniques. There are essentially two types of distribution models, parametric and non-parametric. The former assumes that data distribution follows a known parametric model, while in the latter there is no *a priori* assumption about the data distribution. In this dissertation, we make use of a parametric model, more specifically a Gaussian mixture model, to represent the candidate solutions of a MOO method. This model is formed by a composition of Gaussian functions, each representing a member of the population in the decision space. The model parameters are obtained from the information provided by an MCDM technique, the TOPSIS algorithm. New candidate solutions are sampled from this model using a special scheme that will be detailed in the next chapter, where we formally present our method.

5 The proposed method

In this chapter, we formally detail the proposed method, denoted by MOMCEDA (*Multi-Objective Multi-Criteria Estimation of Distribution Algorithm*) (SOUSA BEZERRA *et al.*, 2018). We propose a hybrid algorithm, composed of EMOA and MCDM techniques to solve MOO problems. The search is guided by the preference information established *a priori* by a decision maker, which is used by the MCDM technique to rank the solutions and build a distribution model of the population, such that promising regions of the search space are associated with higher probabilities of sampling new candidate solutions. This way, we expect the method to produce better solutions that will improve its performance in multi-objective optimization problems.

The MCDM technique used by MOMCEDA is the TOPSIS algorithm, introduced in Section 3.2. This technique allows the candidate solutions to be evaluated regarding multiple criteria defined by a decision maker. Each criterion represents a decision maker's preference, and its evaluation represents the extension with which a candidate solution reaches this preference.

In the case of a hybrid technique composed of EMOA and MCDM techniques, like in our method, it is important to distinguish the evaluation of the objective functions from the evaluation of the preference criteria. Although the decision maker can choose to have the objective functions as part of the criteria set, this is not mandatory. In fact, this approach may not be a good idea to maintain population diversity. As explained in Section 3.2, the TOPSIS method will classify the candidate solutions based on their distance to the ideal values of the criteria set. If the objective functions are part of the criteria set, the candidate solutions that are close to the ideal point in the objective space will have a better classification. Unless this behavior is desired by the decision maker, this approach should be avoided, since the non-dominated solutions located far from the ideal point will most likely be discarded during the search.

5.1 MOMCEDA

The main structure of MOMCEDA follows the one proposed on NSGA-II and NSGA-III methods, described in sections 2.3.1 and 2.3.2, respectively. The algorithm has a fixed population size of N candidate solutions that are submitted to an evolutionary search process until it reaches a stopping condition. This condition can be based on a maximum number of

generations, on a maximum number of objective functions evaluations, on desired values of performance metrics, among other options.

The steps performed by MOMCEDA are summarized in Algorithm 5. In lines 1 and 2, we set the counter t of generations to its initial value and randomly initialize the population of candidate solutions P_t and the offspring population Q_t. The main loop of the algorithm comprises the steps from lines 4 to 16, corresponding to one generation.

Algorithm 5: MOMCEDA

Result: Non-dominated individuals of P_t

1 $t = 1$;
2 P_t, Q_t = initialize-population(N);
3 **while** *stopping condition not satisfied* **do**
4 $\quad R_t = P_t \cup Q_t$;
5 $\quad \mathcal{F}$ = fast-non-dominated-sort(R_t);
6 $\quad P_{t+1} = \emptyset$;
7 $\quad$ **while** $|P_{t+1}| \leq N$ **do**
8 $\quad\quad$ evaluate-criteria $(R_t, \mathcal{F})$;
9 $\quad\quad \mathcal{L}$ = TOPSIS(R_t);
10 $\quad\quad P_{t+1} = P_{t+1} \cup \mathcal{L}[1]$;
11 $\quad\quad R_t = R_t \backslash (\mathcal{L}[1])$;
12 $\quad$ **end**
13 $\quad \mathcal{L}'$ = TOPSIS(P_{t+1});
14 $\quad \mathcal{M}$ =mixture-model($P_{t+1}, \mathcal{L}'$);
15 $\quad Q_{t+1}$ = offspring($P_{t+1}, \mathcal{M}, \alpha, p_m, \boldsymbol{\sigma}_m$);
16 $\quad t = t + 1$;
17 **end**

In line 5, we use the non-domination sorting process, described in Section 2.3.1, to classify all the candidate solutions from both populations (represented by the unified population R_t) into non-domination levels. In line 6, we initialize the next generation's population P_{t+1}, which will be filled in the steps performed in the loop from lines 8 to 11. In line 8, we update the evaluation of the remaining candidate solutions in R_t regarding the preference criteria set. The adopted criteria set will be detailed in the next section. In line 9, the TOPSIS method is called to sort these candidate solutions in a list $\mathcal{L}$ according to the criteria values. In line 10, the best individual obtained from the TOPSIS result is added to the next generation's population P_{t+1}, and in line 11 it is removed from the current pool R_t of individuals being evaluated. Note that removing members from this pool may change the evaluation of the criteria for its remaining members. This is the reason why individuals are added to P_{t+1} one at a time. Depending on the criteria set, however, the criteria values may not be affected by changes in R_t, and will not need to be re-evaluated after a member is removed. In this

case, the first N members of the sorted list $\mathcal{L}$ can be directly added to P_{t+1} at once.

Once P_{t+1} is filled with the best N members from R_t, we call the TOPSIS method in line 13 one more time to sort all the members of the new population. In line 14, the sorted list of members is used to derive a distribution model $\mathcal{M}$ in the search space, more specifically a Gaussian mixture model, as described in Section 4.2. In line 15, new solutions are sampled from this model to originate a new offspring population Q_{t+1}. The steps performed by the offspring function are detailed in Section 5.4. Finally, the counter t of generations is incremented in line 16.

5.2 The preference criteria

A distinct aspect of our approach when compared to other hybrid techniques is that the TOPSIS algorithm allows the use of multiple preference approaches. Most MOO metaheuristics sorts their solutions according to only one preference criterion at a time. Since non-dominated solutions are desired in this kind of problem, the non-domination level is the most important criterion to be observed. Usually, another criterion is evaluated to sort solutions that are non-dominated among each other. This criterion varies according to the approach. For instance, NSGA-II, NSGA-III and SPEA2 favor candidate solutions which contribute the most to improve the population diversity, while SMS-EMOA uses a measure of the solution's contribution to the hypervolume indicator. In a type of reference point based approach, non-dominated solutions are compared regarding how close they are to the user defined aspiration levels. In our method, however, it is possible to simultaneously include all these preference criteria, or as many as the decision maker wishes. It is up to him/her to decide which one is more important than the other, and express that in the form of a weighting vector responsible for defining the relative relevance of the criteria.

As mentioned in Section 3.1, the use of MCDM techniques in *a priori* approaches depends on the decision maker's knowledge about the problem. Here, we took the criteria used by the previously mentioned algorithms as inspiration to define the set of preferences used by our method. The first and most important criterion is the non-domination level, for obvious reasons. The next two criteria evaluate the population's diversity by recurring to the NSGA-III reference-point-based approach, introduced in Section 2.3.2: the second criterion measures the neighborhood size $\rho(\pi(\mathbf{s}))$ of the closest reference direction $\pi(\mathbf{s})$ of an individual $\mathbf{s}$ in the normalized space, and the third criterion is the Euclidean distance $d(\mathbf{s})$ between $\mathbf{s}$ and $\pi(\mathbf{s})$. Since we want to maintain the population's diversity, minimizing these last two criteria is desired. The fourth and final criterion is the solution's contribution to the hypervolume indicator, as in the SMS-EMOA algorithm introduced in Section 2.3.3. This is

a maximization criterion, so we negate it to obtain another minimization criterion.

5.3 The Gaussian mixture model

The distribution model adopted by MOMCEDA to represent the candidate solutions in the decision space is a Gaussian mixture model, as explained in Section 4.2. Each candidate solution is the center of a Gaussian function, whose weighting coefficient π_k is proportional to the relative quality of that solution. The relation between the weighting coefficient and the position of its corresponding candidate solution in the ranking follows a predefined function, so that the coefficient is as high as the relevance of its corresponding candidate solution. In this research, we adopted three kinds of decreasing functions: linear, exponential and logarithmic, represented in equations (5.1), (5.2) and (5.3), respectively:

$$\begin{cases} \pi_k = A \cdot k + B \\ \pi_N = K_{lin} \end{cases} \tag{5.1}$$

$$\begin{cases} \pi_k = C \cdot D^{-k} \\ \dfrac{\pi_N}{\pi_1} = K_{exp} \end{cases} \tag{5.2}$$

$$\begin{cases} \pi_k = E \cdot log(N - k + F) \\ \pi_N = K_{log} \end{cases} \tag{5.3}$$

where k is the position of the candidate solution in the sorted list provided by the TOPSIS algorithm, A, B, C, D, E and F are constants, determined by substituting equations (5.1), (5.2) and (5.3) in Eq. (4.1), and K_{lin}, K_{exp} and K_{log} are initial conditions defined by the user.

For $K_{lin} = 0$, we have:

$$\begin{gathered} \pi_N = 0 \Leftrightarrow AN + B = 0 \Leftrightarrow B = -AN \\ \therefore \pi_k = A(k - N) \\ \sum_{k=1}^{N} A(k - N) = 1 \Leftrightarrow A \sum_{k=1}^{N} (k - N) = 1 \\ \Leftrightarrow A \left[\tfrac{N(N+1)}{2} - N^2\right] = 1 \Leftrightarrow A \left[\tfrac{N - N^2}{2}\right] = 1 \\ \Leftrightarrow A = \tfrac{2}{N(1-N)} \end{gathered}$$

$$\therefore \pi_k = \frac{2(k - N)}{N(1 - N)} \tag{5.4}$$

For $K_{exp} = \gamma$, we have:

$$\pi_N = \gamma \pi_1 \Leftrightarrow CD^{-N} = \gamma CD^{-1} \Leftrightarrow D = \gamma^{\frac{-1}{N-1}}$$

$$\therefore \pi_k = C \cdot \gamma^{\frac{k}{N-1}}$$
$$\sum_{k=1}^{N} C \cdot \gamma^{\frac{k}{N-1}} = 1 \Leftrightarrow C \sum_{k=1}^{N} (\gamma^{\frac{1}{N-1}})^k = 1$$
$$\Leftrightarrow C \cdot \gamma^{\frac{1}{N-1}} \cdot \frac{(1-\gamma^{\frac{N}{N-1}})}{(1-\gamma^{\frac{1}{N-1}})} = 1 \Leftrightarrow C = \frac{1-\gamma^{\frac{1}{N-1}}}{\gamma^{\frac{1}{N-1}}(1-\gamma^{\frac{N}{N-1}})}$$
$$\therefore \pi_k = \frac{(1-\gamma^{\frac{1}{N-1}})}{(1-\gamma^{\frac{N}{N-1}})} \cdot \gamma^{\frac{k-1}{N-1}} \tag{5.5}$$

For $K_{log} = 0$, we have:

$$\pi_N = 0 \Leftrightarrow E \log(F) = 0 \Leftrightarrow F = 1$$
$$\therefore \pi_k = E \cdot log(N-k+1)$$
$$\sum_{k=1}^{N} E \cdot log(N-k+1) = 1 \Leftrightarrow E \sum_{k=1}^{N} log(N-k+1) = 1$$
$$\Leftrightarrow E \cdot log\left(\prod_{k=1}^{N}(N-k+1)\right) = 1 \Leftrightarrow E = \frac{1}{log(N!)}$$
$$\therefore \pi_k = \frac{log(N-k+1)}{log(N!)} \tag{5.6}$$

Figure 8 shows a graphical representation of the three types of decreasing functions adopted for the weighting coefficients, given by Equations (5.4), (5.5) and (5.6), for $N = 100$ and $\gamma = 10^{-3}$. In this case, the exponential function has the largest decreasing rate, emphasizing the first members of the sorted list. On the other side, the logarithmic function has the smallest decreasing rate, penalizing only the last members of the sorted list. The linear function is between the other two.

Once the mixture coefficients are determined, new solutions can be sampled. Traditionally, sampling data from a Gaussian mixture model is a simple task: first, one of the N Gaussian functions is selected by a process similar to a roulette wheel operator: each π_k coefficient corresponds to a portion of the interval $[0, 1]$; it suffices to generate a uniformly distributed random number in this interval to find out which Gaussian function will be used. Then, this function is taken to actually generate the new data. In MOMCEDA, however, we adopted a modified sampling scheme, where new solutions are sampled in pairs, and the Gaussian functions parameters are not defined *a priori*. This scheme will be detailed in the next section.

5.4 Offspring generation

As mentioned in Section 4.2, MOMCEDA generates a new offspring population by applying refreshing operators responsible for perturbing new individuals sampled by the

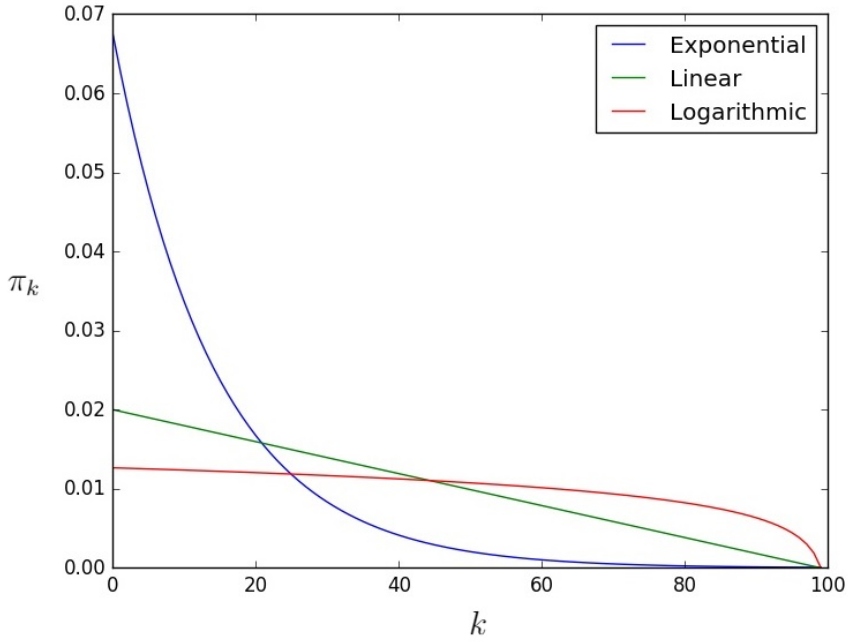

Figure 8 – Graphical representation of the three types of decreasing functions adopted for the weighting coefficients for $N = 100$ and $\gamma = 10^{-3}$.

Gaussian mixture model. These operators are used to increase the chance of reaching unexplored areas of the search space while simultaneously exploring promising regions detected by already found candidate solutions.

Many established methods, such as NSGA-II and NSGA-III, adopt an offspring generation scheme based on the use of crossover and mutation operators that are applied to selected individuals from the population. The Simulated Binary Crossover, or SBX operator (DEB; AGRAWAL, 1994) is a widely used operator on real-coded Genetic Algorithms (GA's) due to its efficiency and search power. The search power of a crossover operator is defined in terms of the probability of creating an arbitrary child solution from a given pair of parent solutions. In other words, it is a measure of how flexible the operator is in creating an arbitrary point in the search space. Deb and Agrawal defined a spread factor β as the ratio of the absolute differences of the children points $\boldsymbol{c}^{(1)}, \boldsymbol{c}^{(2)}$ to that of the parent points $\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}$:

$$\beta = \left| \frac{\boldsymbol{c}^{(1)} - \boldsymbol{c}^{(2)}}{\boldsymbol{x}^{(1)} - \boldsymbol{x}^{(2)}} \right| \tag{5.7}$$

The SBX operator was proposed to simulate the search power of single-point crossover used in binary-coded GA's. Then, the probability distribution of β for the SBX operator should be similar to the one for the single-point crossover, which leads to higher occurrences

of values of β close to 1, meaning that the average distance of children points tends to be close to the average distance of parent points. This implies an interesting property: as the search progresses and the population members become closer in the decision space, the crossover operator will also generate children that are closer among them, refining the search.

Motivated by the properties of this operator, we decided to define the standard deviation of the Gaussian functions from the mixture model in terms of the distance between individuals in the decision space. The steps performed by the offspring function are summarized in Algorithm 6, which takes as input a population P of N candidate solutions and its distribution model $\mathcal{M}$ and produces as output an offspring population Q of the same size. We sample new individuals in pairs. From lines 3 to 7, the roulette wheel operator is executed twice or more (as many times as necessary) to produce two distinct individuals $\boldsymbol{x}^{(1)}$ and $\boldsymbol{x}^{(2)}$ from the mixture model. Then, in lines 9 and 10, two new individuals $\boldsymbol{c}^{(1)}$ and $\boldsymbol{c}^{(2)}$ are sampled from the Gaussian functions centered at $\boldsymbol{x}^{(1)}$ and $\boldsymbol{x}^{(2)}$. We assume there is no correlation between the variables in the decision space so that the standard deviation σ_i for the i-th decision variable is given by:

$$\sigma_i = \alpha|x_i^{(1)} - x_i^{(2)}| \tag{5.8}$$

where α is a proportionality constant to be defined. This constant has a relation with the spread parameter β, since the standard deviation affects the spread of the generated individuals. For a random variable $\mathcal{X}$ that follows a normal distribution $\mathcal{N}(\mu, \sigma)$, the 95% confidence interval of $\mathcal{X}$ is approximately given by:

$$\mu - 1.96\sigma \leq \mathcal{X} \leq \mu + 1.96\sigma \tag{5.9}$$

Let $\Delta(c_i) = c_i^{(1)} - c_i^{(2)}$, where $c_i^{(k)} \sim \mathcal{N}(x_i^{(k)}, \sigma_i), k = \{1, 2\}$. The variance of $\Delta(c_i)$, denoted $\mathrm{Var}[\Delta(c_i)]$, is given by:

$$\mathrm{Var}[\Delta(c_i)] = \mathrm{Var}[c_i^{(1)}] + \mathrm{Var}[c_i^{(2)}] - 2\mathrm{Cov}[c_i^{(1)}, c_i^{(2)}] \tag{5.10}$$

where $\mathrm{Var}[c_i^{(k)}] = (\sigma_i)^2$ and $\mathrm{Cov}[c_i^{(1)}, c_i^{(2)}]$ is the covariance between $c_i^{(1)}$ and $c_i^{(2)}$. Since $c_i^{(1)}$ and $c_i^{(2)}$ are independent random variables, $\mathrm{Cov}[c_i^{(1)}, c_i^{(2)}] = 0$ and it follows that $\mathrm{Var}[\Delta(c_i)] = 2(\sigma_i)^2$. Thus $\Delta(c_i) \sim \mathcal{N}(\Delta(x_i), \sqrt{2}\sigma_i)$, where $\Delta(x_i) = x_i^{(1)} - x_i^{(2)}$. Hence, the 95% confidence interval for $\Delta(c_i)$ is given by:

$$\Delta(x_i) - 1.96 \cdot \sqrt{2}\sigma_i \leq \Delta(c_i) \leq \Delta(x_i) + 1.96 \cdot \sqrt{2}\sigma_i \tag{5.11}$$

If $\Delta(x_i) = 0$, the parent variables are equal, and the children will be assigned to these same values, since the standard deviation will be zero. If $\Delta(x_i) < 0$, we can switch the individuals to make $\Delta(x_i)$ positive in this analysis. If $\Delta(x_i) > 0$, then $\Delta(c_i) > 0$ for sufficiently small values of standard deviation, which implies that the children are generated close to their respective parents. Substituting σ_i by Eq. (5.8) in Eq. (5.11), we obtain:

$$\Delta(x_i)(1 - 1.96 \cdot\sqrt{2}\alpha) \leq \Delta(c_i) \leq \Delta(x_i)(1 + 1.96 \cdot\sqrt{2}\alpha) \tag{5.12}$$

Since we are in the case where $\Delta(x_i) > 0$ and $\Delta(c_i) > 0$, we can divide all members of the previous inequations by $\Delta(x_i)$ and substitute β by Eq. (5.7), which gives a 95% confidence relation between α and β:

$$1 - 1.96 \cdot\sqrt{2}\alpha \leq \beta \leq 1 + 1.96 \cdot\sqrt{2}\alpha \tag{5.13}$$

By choosing a sufficient small value for α, we notice from Eq. (5.13) that the spread factor for our sampling scheme will be around 1 in 95% of the cases, similar to the behavior of the SBX operator. Thus, our offspring generation scheme is able to incorporate the advantages of using a distribution model as well as some of the interesting properties of the SBX operator.

To promote refreshing, once the two individuals are generated, a first refreshing operator is applied, which gives a 50% probability of switching each variable between the individuals. This is essentially the uniform crossover operator usually adopted in evolutionary computation. In Algorithm 6, these steps are represented from lines 11 to 15, where n is the number of decision variables and *random-number(0,1)* samples a uniformly distributed random number in the interval $[0, 1]$. After generating all N individuals, we apply a second refreshing operator to each variable of every member with a small probability p_m. It consists of a Gaussian mutation operator, which adds to a decision variable a number obtained from a normal distribution with zero mean and a initial value of standard deviation $\boldsymbol{\sigma}_m$ to be defined. If a bounded variable falls out of range, its value is attributed to the closest bound. After applying the mutation operator in line 22, we verify in line 23 if the new individual is better than the original one - in our case, we adopt the dominance relation to compare individuals. If it is better, then we keep the new individual, as shown in line 24; otherwise, we keep the original one. The standard deviation of the mutation operator is adjusted according to its performance. After five consecutive positive results for this operator, we increase by 10% the standard deviation for the variables involved in the mutation; after five consecutive negative results, we reduce it by 10%. This is represented by the *update-std-dev(σ_j)* function in line 26. The refreshing operators are adopted to help the search reach other potentially promising regions of the search space not yet mapped by the distribution model.

Algorithm 6: offspring($P, \mathcal{M}, \alpha, p_m, \boldsymbol{\sigma}_m$)

Result: Offspring population Q

```
1  Q = ∅;
2  while |Q| ≤ N do
3      x^(1) = roulette-wheel(P, M);
4      x^(2) = roulette-wheel(P, M);
5      while x^(1) = x^(2) do
6          x^(2) = roulette-wheel(M);
7      end
8      σ = α|x^(1) − x^(2)|;
9      c^(1) = sample-Gaussian-function(x^(1), σ);
10     c^(2) = sample-Gaussian-function(x^(2), σ);
11     for i = 1; i ≤ n do
12         if random-number(0,1) ≤ 0.5 then
13             switch(c_i^(1), c_i^(2));
14         end
15     end
16     Q = Q ∪ {c^(1), c^(2)};
17 end
18 for i = 1; i ≤ N do
19     for j = 1; j ≤ n do
20         if random-number(0,1) ≤ p_m then
21             σ = σ_m;
22             x = sample-Gaussian-function(Q[i]_j, σ_j);
23             if x is better than Q[i]_j then
24                 Q[i]_j = x;
25             end
26             σ_j = update-std-dev(σ_j);
27         end
28     end
29 end
```

5.5 Time Complexity of MOMCEDA

The non-dominated sorting (line 5 in Algorithm 5) of a population of size $2N$ having M-dimensional objective vectors requires $O(N^2M)$ computations. The evaluation of criteria (line 8) requires $O(N^2M)$ computations for NSGA-III criteria (DEB; JAIN, 2013). The evaluation of the hypervolume contributions is the bottleneck of the algorithm, since the time complexity to calculate the hypervolume indicator grows exponentially with the number of objectives. However, for $M \leq 3$, there are faster methods to directly calculate the hypervolume contributions, requiring $O(N \log N)$ computations (BEUME *et al.*, 2007). The TOPSIS

method (lines 9 and 13) requires $O(MN)$ computations. Sorting individuals to build the mixture model (line 14) requires $O(N \log N)$ computations. The offspring generation (line 15) requires $O(N)$ computations. Hence, the overall worst-case complexity of one generation of MOMCEDA is $O(N^2M)$ for $M \leq 3$.

5.6 Summary

In this chapter, we presented the proposed method, denoted by MOMCEDA (*Multi-Objective Multi-Criteria Estimation of Distribution Algorithm*), which follows the structural conception of NSGA-II, also adopted in NSGA-III. Here, some of the preference criteria adopted by state-of-the-art EMOAs, including non-domination level, diversity measures and contribution to the hypervolume indicator, are taken together as inputs to a multi-criteria decision making (MCDM) strategy, more specifically the TOPSIS algorithm, responsible for sorting all candidate solutions. New candidate solutions are then generated from this sorted list using a Gaussian mixture model. Essentially, a roulette wheel is used to choose a pair of the current candidate solutions according to the relative quality, suitably determined by TOPSIS, and after that a pair of new candidate solutions is generated as Gaussian perturbations centered at the corresponding parents. The standard deviation is defined in terms of the absolute distance of the parents in the decision space, and refreshing operators are applied after the sampling, aiming at reaching other promising regions of the search space not yet mapped by the distribution model. Therefore, the distinctive aspects, when MOMCEDA is compared to NSGA-II and NSGA-III, are the way the current population of candidate solutions is ranked and the strategy adopted to generate new individuals. Table 1 summarizes MOMCEDA's parameters.

Table 1 – MOMCEDA's parameters.

Parameter	**Description**
N	Population size
$\boldsymbol{w}$	Relative relevance of the user preferences
α	Mixture model parameter
p_m	Mutation probability
$\boldsymbol{\sigma}_m$	Mutation parameter

6 Computational experiments

In this chapter, we present the results of some computational experiments executed to evaluate MOMCEDA's performance in distinct MOO scenarios. The results analysis is based on the values of commonly used performance metrics in the field, comparing MOMCEDA with other state-of-the-art methods. Part of the results on this chapter were based on SOUSA BEZERRA *et al.* (2018). We implemented our method using Python 2.7.12, with a version available at the repository *https://github.com/pedro-mariano/MOMCEDA*.

6.1 Test problems

6.1.1 Two-on-One problems

The Two-on-One is a set of problems with two objectives, consisting of a polynomial function f_1 of degree four with two optima, and the sphere function f_2, which is of degree two, as shown in Eq. (6.1) (PREUSS *et al.*, 2006):

$$\min \begin{cases} f_1(\boldsymbol{x}) = x_1^4 + x_2^4 - x_1^2 + x_2^2 - cx_1x_2 + dx_1 + 20 \\ f_2(\boldsymbol{x}) = (x_1 - k)^2 + (x_2 - l)^2 \end{cases} \tag{6.1}$$

where $x_i \in [-3, 3] \quad \forall i \in \{1, 2\}$.

The level (niveau) of the optima of f_1 can be adjusted smoothly via parameter d. With parameter $c = 0$, both minima are located at the x_1 axis, but for increasing c, their connecting line is rotated counterclockwise, until its gradient is nearly 1. Function f_2 is unimodal and the location of its minimum is determined by parameters k and l. For $k = l = 0$ it is located at the origin, right between the minima of the bimodal function f_1. By variation of k and l the minimum is moved away from the connecting line of the minima of f_1. Besides changing the Pareto front, this also affects the Pareto set.

6.1.2 ZDT problems

The classical ZDT test benchmark was originally proposed in ZITZLER *et al.* (2000), and is a widely used test set to compare MOO methods. The set contains six test problems

with two objectives and different characteristics, and has the following structure:

$$\min \begin{cases} f_1(\boldsymbol{x}) \\ f_2(\boldsymbol{x}) = g(\boldsymbol{x}) \cdot h(f_1(\boldsymbol{x}), g(\boldsymbol{x})) \end{cases} \tag{6.2}$$

The functions $f_1(\boldsymbol{x}), g(\boldsymbol{x})$ and $h(\boldsymbol{x})$ define the different characteristics for each problem. They feature different Pareto front conformations, such as convex, non-convex, disconnected and non-uniformly spaced, as illustrated in Fig. 9. For all problems, except ZDT5, which is defined for binary variables and is not used in this work, the Pareto optimal solutions are such that $g(\boldsymbol{x}) = 1$ (COELHO, 2011). Next, the problems will be detailed.

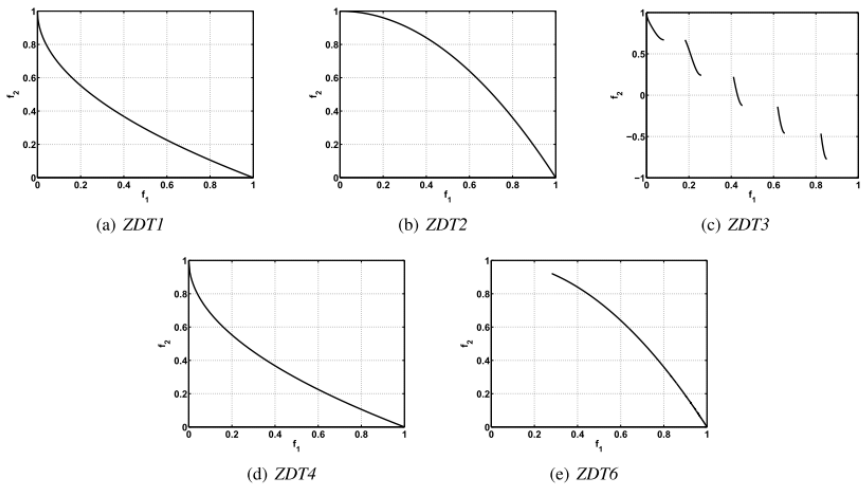

Figure 9 – Graphical representation of the Pareto front for each ZDT test problem used in this work.

6.1.2.1 ZDT1

The ZDT1 problem has $n = 30$ variables belonging to the domain $x_i \in [0; 1], i = 1, ..., n$. It figures a continuous, convex Pareto front with uniformly distributed solutions (see Fig. 9.(a)). Its functions are represented in Eq. (6.3). The Pareto optimal decision variables are such that $0 \leq x_1^* \leq 1$ and $x_i^* = 0$ for $i = 2, ..., n$.

$$\begin{cases} f_1(\boldsymbol{x}) = x_1 \\ g(\boldsymbol{x}) = 1 + \dfrac{9}{n-1} \displaystyle\sum_{i=2}^{n} x_i \\ h(f_1(\boldsymbol{x}), g(\boldsymbol{x})) = 1 - \sqrt{\dfrac{f_1(\boldsymbol{x})}{g(\boldsymbol{x})}} \end{cases} \tag{6.3}$$

6.1.2.2 ZDT2

The ZDT2 problem also has $n = 30$ variables belonging to the domain $x_i \in [0; 1], i = 1, ..., n$, but now it figures a continuous, non-convex Pareto front with uniformly distributed solutions (see Fig. 9.(b)). Its functions are represented in Eq. (6.4). The Pareto optimal decision variables are such that $0 \leq x_1^* \leq 1$ and $x_i^* = 0$ for $i = 2, ..., n$.

$$\begin{cases} f_1(\boldsymbol{x}) = x_1 \\ g(\boldsymbol{x}) = 1 + \dfrac{9}{n-1} \displaystyle\sum_{i=2}^{n} x_i \\ h(f_1(\boldsymbol{x}), g(\boldsymbol{x})) = 1 - \left(\dfrac{f_1(\boldsymbol{x})}{g(\boldsymbol{x})} \right)^2 \end{cases} \tag{6.4}$$

6.1.2.3 ZDT3

The ZDT3 problem also has $n = 30$ variables belonging to the domain $x_i \in [0; 1], i = 1, ..., n$, but this time it figures a discontinuous Pareto front formed by five segments (see Fig. 9.(c)). Its functions are represented in Eq. (6.5). The Pareto front is defined by points in $x_i^* = 0$ for $i = 2, ..., n$ and some points in $0 \leq x_1^* \leq 1$.

$$\begin{cases} f_1(\boldsymbol{x}) = x_1 \\ g(\boldsymbol{x}) = 1 + \dfrac{9}{n-1} \displaystyle\sum_{i=2}^{n} x_i \\ h(f_1(\boldsymbol{x}), g(\boldsymbol{x})) = 1 - \sqrt{\dfrac{f_1(\boldsymbol{x})}{g(\boldsymbol{x})}} - \left(\dfrac{f_1(\boldsymbol{x})}{g(\boldsymbol{x})} \right) \cdot \sin(10 \cdot \pi \cdot f_1(\boldsymbol{x})) \end{cases} \tag{6.5}$$

6.1.2.4 ZDT4

The ZDT4 problem has $n = 10$ variables, whose domain is defined by $x_1 \in [0; 1]$ and $x_i \in [-5; 5], i = 2, ..., n$. It figures a continuous, convex Pareto front (see Fig. 9.(d)), but this problem has a large number of Pareto local optimal solutions, corresponding to $0 \leq x_1^* \leq 1$ and $x_i^* = 0.5m$ for $i = 2, ..., n$, where m is an integer number in the interval $[-10, 10]$. Its functions are represented in Eq. (6.6). The Pareto optimal decision variables are such that

$0 \leq x_1^* \leq 1$ and $x_i^* = 0$ for $i = 2, ..., n$.

$$\begin{cases} f_1(\boldsymbol{x}) = x_1 \\ g(\boldsymbol{x}) = 1 + 10(n-1) + \sum_{i=2}^{n} [x_i^2 - 10 \cdot \cos(4 \cdot \pi \cdot x_i)] \\ h(f_1(\boldsymbol{x}), g(\boldsymbol{x})) = 1 - \sqrt{\frac{f_1(\boldsymbol{x})}{g(\boldsymbol{x})}} \end{cases} \tag{6.6}$$

6.1.2.5 ZDT6

The ZDT6 problem has $n = 10$ variables belonging to the domain $x_i \in [0; 1], i = 1, ..., n$. It figures a continuous, non-convex Pareto front with a non-uniform distribution of solutions (see Fig. 9.(e)). Its functions are represented in Eq. (6.7). The Pareto optimal decision variables are such that $0 \leq x_1^* \leq 1$ and $x_i^* = 0$ for $i = 2, ..., n$.

$$\begin{cases} f_1(\boldsymbol{x}) = 1 - \exp(-4x_1) \cdot \sin^6(6 \cdot \pi \cdot x_1) \\ g(\boldsymbol{x}) = 1 + 9 \cdot \left(\frac{\sum_{i=2}^{n} x_i}{9} \right)^{0.25} \\ h(f_1(\boldsymbol{x}), g(\boldsymbol{x})) = 1 - \left(\frac{f_1(\boldsymbol{x})}{g(\boldsymbol{x})} \right)^2 \end{cases} \tag{6.7}$$

6.2 Performance metrics

To analyze MOMCEDA's performance on the test problems and compare it with other state-of-the-art algorithms, we adopted three performance metrics. The first one is the hypervolume indicator, introduced in Section 2.3.3 which should be maximized. This metric rewards the convergence towards the Pareto front as well as the representative distribution of points along the front.

The second metric is the convergence measure (DEB *et al.*, 2003). When the exact knowledge of the Pareto front is available, we calculate B uniformly distributed (on the $f_1 - f_2 - \cdots - f_{M-1}-$plane) solutions $\mathbf{P}^*$ at the Pareto front. For each point in the $(M-1)-$dimensional plane, f_M is calculated from the known Pareto front description. Then, the Euclidean distance of each obtained solution from the nearest solution in $\mathbf{P}^*$ is computed. The average of the distance value of all obtained solutions is defined as the convergence measure here. It is a metric to be minimized.

The third metric is the Inverse Generational Distance (IGD metric) (DEB; JAIN, 2013), a metric which can provide a combined information about the convergence and diversity of the obtained solutions. When reference points or reference directions are supplied, and the exact Pareto-optimal surface is known, we can exactly locate the targeted Pareto-optimal points. We compute these targeted points and call them a set $\mathbf{Z}$. For any algorithm, we obtain the final non-dominated points in the objective space and call them the set $\mathbf{A}$. Now, we compute the IGD metric as the average Euclidean distance of points in set $\mathbf{Z}$ with their nearest members of all points in set A, as expressed in Eq. (6.8), where $\boldsymbol{z}_i \in \mathbf{Z}, \boldsymbol{a}_j \in \mathbf{A}$. A set with a smaller value for the IGD metric is better.

$$IGD(\mathbf{A}, \mathbf{Z}) = \frac{1}{|\mathbf{Z}|} \sum_{i=1}^{|\mathbf{Z}|} \min_{j=1}^{|\mathbf{A}|} \|\boldsymbol{z}_i, \boldsymbol{a}_j\|_2 \tag{6.8}$$

6.3 Results

6.3.1 Distribution analysis

To analyze and understand how the distribution model operates in the search process, we tested MOMCEDA on the Two-on-One problem with $d = l = k = 0$ and the parameter settings listed in Table 2, where n is the number of decision variables of the test problem. The array $\boldsymbol{w}$ of relative relevance of the user preferences follows the criteria order presented in Section 5.2. We adopted the decreasing linear weighting coefficient function of Eq. (5.6). More details about the choice of parameters will be provided in the next sections. The objective functions of this problem are bi-dimensional, which means that we are able to visualize the probability density function of the distribution model built by MOMCEDA to represent the population variables in different moments of the search. Since the Pareto set of the Two-on-One problem is known, we can analyze the evolution of the distribution model throughout the search.

The pdf of a Gaussian mixture model can be easily obtained, since it is formed by a composition of Gaussian functions, each one with a weighting coefficient. For a given point (x_1, x_2) of the decision space, we calculate the mixture's pdf value by adding the values from the pdf's of each component of the model and multiplying them by their corresponding coefficient. In the case of our method, however, new points are sampled in pairs and the components' pdf parameters depend on the chosen individuals. Given two components $\mathcal{N}_i$, $\mathcal{N}_j$, with coefficients c_i, c_j, the probability P_{ij} that these two components are chosen together

Table 2 – Parameter settings for MOMCEDA.

Parameter	**Value**	**Description**
N	100	Population size
$\boldsymbol{w}$	$[10, 5, 3, 1]^T$	Relative relevance of the user preferences
α	0.025	Mixture model parameter
p_m	$1/n$	Mutation probability
$\boldsymbol{\sigma}_m$	$0.5(\max(\boldsymbol{x}) - \min(\boldsymbol{x}))$	Mutation parameter
p	$N - 1$	Number of divisions of each objective axis
Neval	20000	Evaluation budget

by two independent executions of the roulette wheel operator is given by:

$$P_{ij} = \begin{cases} \dfrac{P(\mathcal{N}_i, \mathcal{N}_j)}{P\{\bigcup\limits_{\forall m,n \in \{1,\ldots,N\}} (\mathcal{N}_m, \mathcal{N}_n)\}} = \dfrac{2c_i c_j}{1 - \sum\limits_{k=1}^{N} c_k^2}, \text{if } i \neq j \\ 0, \text{if } i = j \end{cases} \tag{6.9}$$

We also must take into account the fact that each variable has a 50% probability of being switched between the two sampled individuals. Let $(x_1^{(i)}, x_2^{(i)})$ and $(x_1^{(j)}, x_2^{(j)})$ be the coordinates of two chosen individuals i and j, respectively, in the decision space. After applying the refreshing operator that switches variables, the new pair of points can be seen as being sampled by the original pair of Gaussian components, centered at $(x_1^{(i)}, x_2^{(i)})$ and $(x_1^{(j)}, x_2^{(j)})$, or by the components centered at $(x_1^{(j)}, x_2^{(i)})$ and $(x_1^{(i)}, x_2^{(j)})$ in case the variables were switched. Let $\mathcal{N}_{ii}$, $\mathcal{N}_{jj}$, $\mathcal{N}_{ji}$ and $\mathcal{N}_{ij}$ represent respectively each one of these normal components. Each component has a chance of 25% of generating new points. We can now express the final pdf value $f(x_1, x_2)$ for our mixture model:

$$f(x_1, x_2) = \sum_{i=1}^{N} \sum_{j=1}^{N} 0,25 P_{ij} [\mathcal{N}_{ii}(x_1, x_2) + \mathcal{N}_{jj}(x_1, x_2) + \mathcal{N}_{ji}(x_1, x_2) + \mathcal{N}_{ij}(x_1, x_2)] \tag{6.10}$$

where $\mathcal{N}_{ij}(x_1, x_2)$ is given by:

$$\mathcal{N}_{ij}(x_1, x_2) = \frac{1}{2\pi\sigma_1\sigma_2} \exp\left\{ \frac{-1}{2} \left[\left(\frac{(x_1 - x_1^{(i)})}{\sigma_1} \right)^2 + \left(\frac{(x_2 - x_2^{(j)})}{\sigma_2} \right)^2 \right] \right\} \tag{6.11}$$

and the other normal components are defined similarly. The standard deviations σ_1 and σ_2 are calculated as described in Section 5.4:

$$\begin{cases} \sigma_1 = \alpha |\boldsymbol{x_1}^{(i)} - \boldsymbol{x_1}^{(j)}| \\ \sigma_2 = \alpha |\boldsymbol{x_2}^{(i)} - \boldsymbol{x_2}^{(j)}| \end{cases} \tag{6.12}$$

The pdf function from Eq. 6.10 was evaluated for the whole decision space. We can see the graphical representation of its values in Fig. 10 in three different moments of the search: at the beginning (Fig. 10a), at the middle (Fig. 10b) and at the end (Fig. 10c). The figures on the right show a representation of the contour lines of the figures on the left. Regions of the decision space close to the peaks have higher chances of generating a new individual. The black points represent a sampling of N individuals from each distribution at the given moment. By looking at the three figures, we can see the evolution of the search from an apparently random behavior in Fig. 10a to a refined search in Fig. 10c. In Fig. 11, we see a graphical representation of the Pareto set for the Two-on-One problem, obtained with two different methods in PREUSS *et al.* (2006). We can see from Figure 10 that the distribution model evolves to set higher probabilities for individuals located close to the line that represents the Pareto set, which is an expected behavior. We also see a peak in Fig. 10c that is much higher than the others, indicating a concentration of high ranked candidate solutions in that region.

6.3.2 Weighting coefficients function analysis

In Section 5.3, we introduced the three types of decreasing functions, linear, exponential and logarithmic, adopted by MOMCEDA to represent the relation between the position k of a member in the sorted list provided by the TOPSIS algorithm and the weighting coefficient π_k of its corresponding Gaussian function of the mixture model. To analyze the impact of the choice of this function, we present here a comparison of the algorithm's performance regarding the final values of the hypervolume indicator for each one of the three functions on the ZDT test problems, using $(1.1, 1.1)$ as reference point for the hypervolume indicator. The parameter settings used are listed in Table 2, where n is the number of decision variables of the test problem. Ten runs were executed for each case, and the results are presented in Table 3. The highlighted values show the function with the best results for the hypervolume indicator in each test problem, and the p-value represents the result of the t-test of significance comparing the highlighted value with the other two.

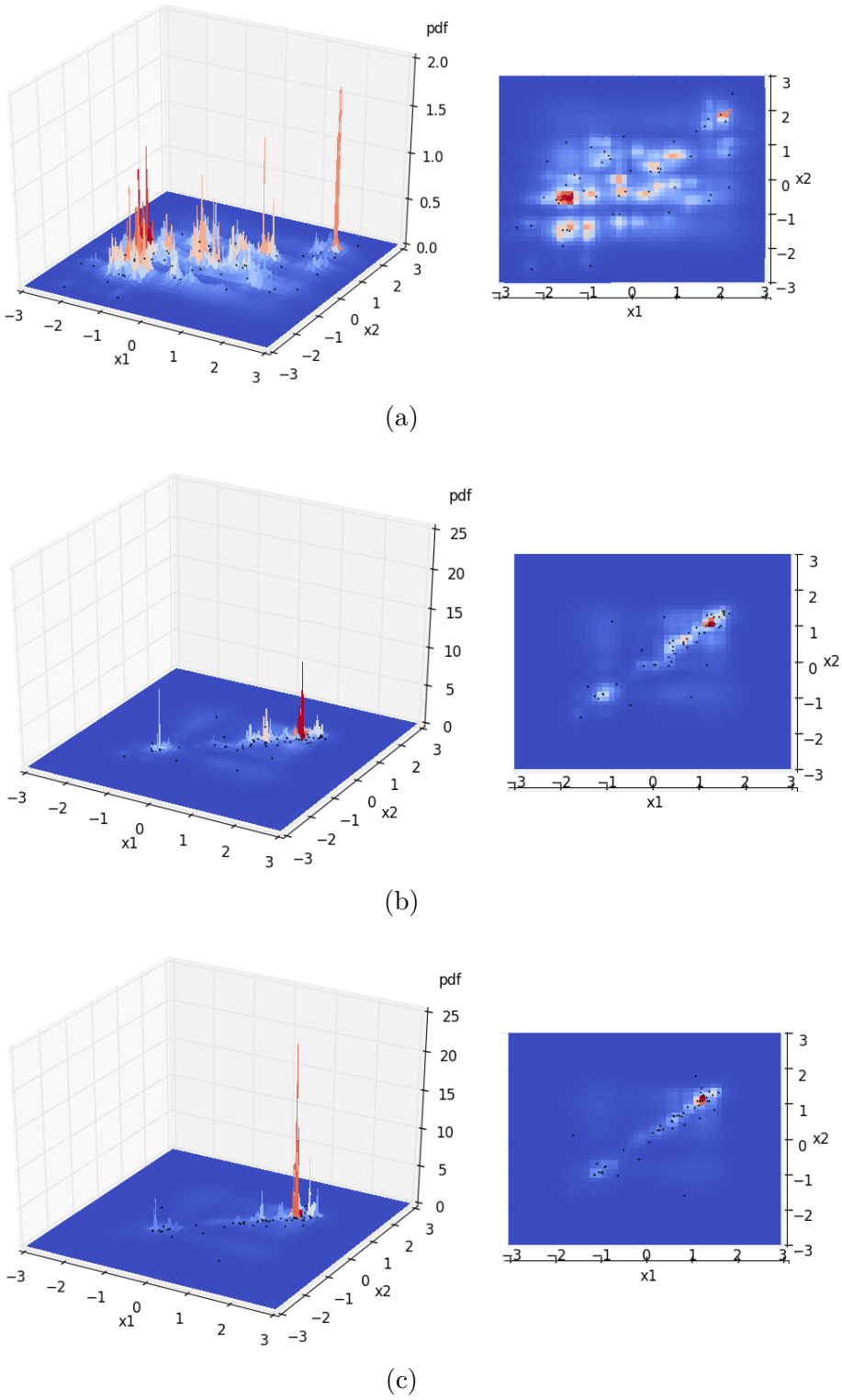

Figure 10 – Evolution of the mixture model's pdf along the seach process for the Two-on-One problem with $d = k = l = 0$.

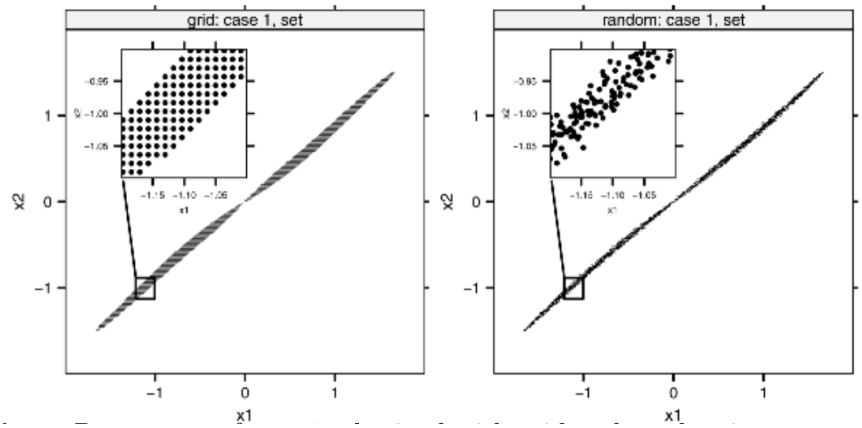

Figure 11 – Graphical representation of the Pareto sets for the Two-on-One problem, obtained with grid and stochastic enumerators (figure extracted from PREUSS *et al.* (2006)).

Table 3 – Hypervolume comparison for different weighting coefficient functions.

Test function	Function	Hypervolume		p-value
		Average	Std. dev.	
ZDT1	Linear	**0.87130**	1.00e-04	-
$n = 30$	Exponential	0.87129	9.26e-05	7.58e-01
	Logarithmic	0.87129	1.84e-04	7.91e-01
ZDT2	Linear	0.53808	4.49e-05	8.88e-08
$n = 30$	Exponential	**0.53827**	4.97e-05	-
	Logarithmic	0.53809	7.13e-05	8.89e-06
ZDT3	Linear	1.32807	1.73e-04	5.18e-03
$n = 30$	Exponential	**1.32833**	1.86e-04	-
	Logarithmic	1.32817	1.80e-04	6.87e-02
ZDT4	Linear	**0.86609**	3.81e-03	-
$n = 10$	Exponential	0.85089	1.71e-02	2.08e-02
	Logarithmic	0.86462	7.67e-03	5.94e-01
ZDT6	Linear	**0.50422**	9.91e-05	-
$n = 10$	Exponential	0.50416	2.88e-04	5.47e-01
	Logarithmic	0.50418	1.49e-04	5.35e-01

Analyzing the statistical results in Table 3 at the 5% level, we see that there is no significant difference between the hypervolume values obtained for the three functions on the

ZDT1 and ZDT6 problems, because the p-values are greater than 0.05. On the ZDT2 and ZDT3 problems, however, the exponential function results are significantly better than the others, except when compared to the logarithmic function on the ZDT3 problem, for which the exponential function is significantly superior only at the 10% level. On the ZDT4 problem, there is no significant difference between the linear function and the logarithmic function, but the linear result is significantly better than the exponential one. The statistical results show that no function is significantly superior at the 5% level to the others in all problems. The exponential function results are the most consistent, since they are significantly better in more cases and worse in less cases than the other functions results, but the linear function results are not far behind. This suggests that as long as the coefficients of the first members of the sorted list are assigned to sufficiently higher values than the last members, the results will not be greatly affected.

6.3.3 Specifying the array of relative relevance of the user preferences

The array $\boldsymbol{w}$ of relative relevance of the user preferences follows the criteria order presented in Section 5.2: non-domination level, neighborhood size, distance to reference direction and exclusive contribution to the hypervolume indicator. To analyze the impact of each criterion on the algorithm's performance, we first performed tests considering four different configurations $\boldsymbol{w}^{(i)}, i = \{1, ..., 4\}$, where $\boldsymbol{w}^{(i)}$ has 1 for the i-th coordinate and 0 for the others, i.e., only the i-th criterion was considered for the TOPSIS ranking. For each configuration, MOMCEDA was executed ten times on the ZDT1 problem using the parameters listed in Table 2, where $n = 30$ is the number of decision variables of the test problem, and the decreasing linear weighting coefficient function from Eq. (5.4) is adopted. Table 4 shows the results for the hypervolume indicator for each configuration, using $(1.1, 1.1)$ as reference point for its calculation. A zero hypervolume value indicates that the non-dominated individuals at the end of the algorithm's execution are dominated by an individual located at this reference point. Since we are not interested in solving a particular problem, but in analyzing the impact of different configurations for $\boldsymbol{w}$, it is not relevant to include results for the other ZDT problems.

The results in Table 4 show that the configurations with a single criterion TOPSIS sorting with the best hypervolume values are $\boldsymbol{w}^{(1)}$ and $\boldsymbol{w}^{(4)}$. As expected, the non-domination level and exclusive contribution to the hypervolume are important criteria regarding the hypervolume metric. However, these criteria alone are not able to promote diversity in the population, which explains why the results are worse than the values obtained before in Table 3. The results for the configurations $\boldsymbol{w}^{(2)}$ and $\boldsymbol{w}^{(3)}$, figuring diversity criteria, indicate that these criteria are not able to promote the approximation of the search towards the Pareto

Table 4 – Hypervolume comparison for different configurations of $\boldsymbol{w}$ on ZDT1 problem, considering only one criterion.

$\boldsymbol{w}$	Hypervolume	
	Average	Std. dev.
$\boldsymbol{w}^{(1)} = [1, 0, 0, 0]^T$	0.86186	1.82e-03
$\boldsymbol{w}^{(2)} = [0, 1, 0, 0]^T$	0.05374	2.75e-02
$\boldsymbol{w}^{(3)} = [0, 0, 1, 0]^T$	0.00000	0.00e00
$\boldsymbol{w}^{(4)} = [0, 0, 0, 1]^T$	0.86965	3.24e-04

front.

Next, we performed tests on the same conditions with different configurations of $\boldsymbol{w}$, but now combining multiple criteria. The results are reported in Table 5. In configurations $\boldsymbol{w}^{(5)}$ and $\boldsymbol{w}^{(6)}$, we tested only the non-domination level and the exclusive contribution to the hypervolume, switching their priorities, and the results were equivalent. Then, from configurations $\boldsymbol{w}^{(7)}$ to $\boldsymbol{w}^{(10)}$, we incorporated the neighborhood size criterion, switching the priorities of the three criteria, but keeping the non-domination level or the exclusive contribution to the hypervolume as high priority criterion. The best results were achieved for the configuration $\boldsymbol{w}^{(10)}$, with the non-domination level having the highest priority, and the exclusive contribution to the hypervolume having the lowest priority. In the sequence, we tested configurations with all four criteria. In configuration $\boldsymbol{w}^{(11)}$, each criterion receives the same weight. From configurations $\boldsymbol{w}^{(12)}$ to $\boldsymbol{w}^{(14)}$, we keep the diversity criteria as second and third priority criteria and keep as high priority criterion the non-domination level or the exclusive contribution to the hypervolume. The configuration $\boldsymbol{w}^{(15)}$ had the best result, with the neighborhood size criterion having higher priority than the distance to the reference direction criterion, and non-domination level as the high priority criterion. A final test was performed for configuration $\boldsymbol{w}^{(16)}$, keeping the same priorities of $\boldsymbol{w}^{(15)}$, but with different values for $\boldsymbol{w}$. There was no significant difference at the 5% level of the significance test comparing the mean hypervolume values of configurations $\boldsymbol{w}^{(15)}$ and $\boldsymbol{w}^{(16)}$ (p-value 0.0506), implying that the priority order of the relative relevance of the criteria is more important than the actual values assigned to $\boldsymbol{w}$.

6.3.4 Refreshing operators analysis

To analyze the impact of the refreshing operators, detailed in Section 5.4, on MOMCEDA's performance, we tested our method in four different situations: without applying any of the refreshing operators, applying one of the two operators each time and applying

Table 5 – Hypervolume comparison for different configurations of $\boldsymbol{w}$ on ZDT1 problem, considering multiple criteria.

$\boldsymbol{w}$	Hypervolume	
	Average	Std. dev.
$\boldsymbol{w}^{(5)} = [2, 0, 0, 1]^T$	0.86993	5.61e-04
$\boldsymbol{w}^{(6)} = [1, 0, 0, 2]^T$	0.86988	2.96e-04
$\boldsymbol{w}^{(7)} = [1, 3, 0, 5]^T$	0.86989	2.88e-04
$\boldsymbol{w}^{(8)} = [3, 1, 0, 5]^T$	0.87008	2.36e-04
$\boldsymbol{w}^{(9)} = [5, 1, 0, 3]^T$	0.87003	3.38e-04
$\boldsymbol{w}^{(10)} = [5, 3, 0, 1]^T$	0.87104	2.60e-04
$\boldsymbol{w}^{(11)} = [1, 1, 1, 1]^T$	0.86998	2.59e-04
$\boldsymbol{w}^{(12)} = [1, 3, 5, 10]^T$	0.86957	4.18e-04
$\boldsymbol{w}^{(13)} = [1, 5, 3, 10]^T$	0.86943	4.01e-04
$\boldsymbol{w}^{(14)} = [10, 3, 5, 1]^T$	0.87098	1.74e-04
$\boldsymbol{w}^{(15)} = [10, 5, 3, 1]^T$	**0.87130**	1.00e-04
$\boldsymbol{w}^{(16)} = [20, 10, 5, 3]^T$	0.87120	1.23e-04

both of them. For each scenario, MOMCEDA was executed ten times on the ZDT1 problem using the parameters listed in Table 2, where $n = 30$ is the number of decision variables of the test problem, and the decreasing linear weighting coefficient function from Eq. (5.4) is adopted. Table 6 shows the results for the hypervolume indicator for each configuration, using $(1.1, 1.1)$ as reference point for its calculation.

Table 6 – Hypervolume comparison on ZDT1 problem, in the presence/absence of refreshing operators.

Refreshing operator(s)	Hypervolume	
	Average	Std. dev.
None	0.05951	6.88e-2
First operator	0.75502	5.87e-02
Second operator	0.86878	6.24e-04
Both	**0.87130**	1.00e-04

The results in Table 6 indicate that, in the absence of refreshing operators, the algorithm's performance was poor. Applying only one of the operators led to better results, but still far from the best possible result, achieved when both of the operators were applied. This outcome reassures the importance of these operators to help the algorithm reach unexplored

promising regions of the search space in order to overcome the difficulty discussed in Section 4.2.

6.3.5 Results on ZDT test problems

In this section, we present the tests performed to compare MOMCEDA's performance with other state-of-the-art algorithms on the ZDT test problems. The parameter settings used to test MOMCEDA on the ZDT problems are listed in Table 2, where n is the number of decision variables for the problem. The choices of parameters values were determined by the best results achieved on the tests described in the previous sections. For the weighting coefficient function, we chose the exponential decreasing relation. We adopted the structured approach presented in Section 2.3.2 to generate the reference points. Since we are dealing with problems with $M = 2$ objective functions, the number H of reference points given by Eq. 2.6 is $H = p + 1$, where p is the number of divisions of each objective axis. We set p so that the number of reference points H is equal to the population size N (DEB; JAIN, 2013).

To make a fair comparison with the results available in the literature, ten runs were executed for each of the test problems, like in BEUME *et al.* (2007). In order to analyze the performance of the proposed algorithm, we firstly present in Figure 12 an average hypervolume evolution comparison between MOMCEDA and the NSGA-II, SPEA2 and SMS-EMOA algorithms for the ZDT1, ZDT2 and ZDT6 problems, using $(1.1, 1.1)$ as reference point for the hypervolume indicator. The competing methods were chosen due to their good performance in practical applications in the literature, and to the ease of access to their implementations. For NSGA-II and SPEA2, we used the implementation from the Python DEAP 1.2.2 framework (FORTIN *et al.*, 2012), and for the SMS-EMOA, the implementation from the Python evoalgos 1.0 package (WESSING, 2017). We did not consider the results for the ZDT3 and ZDT4 in this first comparison due to the occurrence of some unsuccessful runs for the SMS-EMOA implementation.

We also compared the final values of some performance metrics. Part of the results in Table 7 was extracted from YANG *et al.* (2016) and shows a comparison with the NSGA II, C-NSGA-II, NSGA-III, SPEA2, ϵ-MOEA, MOEA/D, SMS-EMOA and SMS-EMOA mdrp algorithms using the hypervolume indicator and the convergence measure like in DEB *et al.* (2003). Finally, Table 8 compares the IGD metric results with the three implementations used to analyze the hypervolume evolution for all problems, except for the ZDT3 problem, since it is non-trivial to calculate the exact location of the targeted points in this case.

The results in Figure 12 show that MOMCEDA outperformed established methods such as NSGA-II and SPEA2 for the three selected problems, and was competitive against

the original SMS-EMOA, which is outperformed on the ZDT6 problem. The evolution of the hypervolume indicator showed a faster convergence towards the Pareto front for the proposed method in the first generations, since it reached higher hypervolume values before the other methods. Besides, the error bars reveal smaller values of standard deviation for MOMCEDA, which means its results are consistent. However, our method was surpassed on later generations by SMS-EMOA on the ZDT1 and ZDT2 problems, implicating that there is room for improvement of solutions at this stage of the search. These results are reinforced in Table 7, where we can see the final values for the hypervolume indicator and the convergence measure. With respect to the hypervolume indicator, the proposed algorithm was outperformed mostly by the versions of SMS-EMOA, which explicitly focus on maximizing this metric. Nevertheless, MOMCEDA is in the top 4 among the 9 compared techniques. It is worth noting that for some methods, such as the SMS-EMOA mdrp (YANG *et al.*, 2016), only the successful runs were used to calculate the metrics for the ZDT4 problem. This is not the case for our algorithm, reinforcing its robustness. Also, MOMCEDA performed well for the ZDT6 problem, whose Pareto front is known to be non-convex and non-uniformly spaced.

As for the convergence measure, our method ranks from the 3rd to the 6th position. This is likely due to the limitations imposed by the NSGA-III criteria used in our algorithm, which tends to give better rankings to the solutions closer to the reference directions; however, these directions are not necessarily equi-spaced along the Pareto front, as with the solutions used to evaluate the convergence measure. This can be verified by looking at the poor results in this metric for the NSGA-III in Table 7. In fact, except for the ZDT2 problem, NSGA-II also performs badly, which can be another reason for MOMCEDA's difficulties with the convergence measure. Meanwhile, the results in Table 8 show how close are the solutions to the Pareto front on the desired reference directions. Here, we can see that MOMCEDA outperformed the other algorithms in most of the cases. There is a performance trade-off on the ZDT6 problem, for which our method performed worse on the IGD metric, but better on the hypervolume and the convergence measures. This problem features a non-uniformly distributed Pareto front, which can explain why methods including diversity measures had worse values of IGD metric than SMS-EMOA.

Finally, regarding MOMCEDA's runtime for the ZDT problems, our implementation took on average 100 seconds for each execution. However, this version is not fully optimized and can be improved to reduce this value. Runtime information is not available on the literature for all the methods analyzed in this Section. MOMCEDA's runtime is likely to be higher due to its nature, but still in the same order of magnitude of its competitors. As shown in sections 2.3.1, 2.3.2 and 5.5, MOMCEDA's time complexity is the same as NSGA-II and

NSGA-III, depending on the case.

6.4 Summary

This chapter was devoted to analyze the performance of the proposed method, MOMCEDA. We first introduced the test problems and the performance metrics that were adopted in this research. We proceeded with an analysis of the distribution model to understand how it operates on the search space, by visualizing the pdf function at different moments of the algorithm's execution. Then, several tests were conducted to analyze the impact of the choices that had to be made, regarding the weighting coefficients function, the definition of the array of relative relevance of the user preferences and the refreshing operators. Finally, the chapter ends with performance comparisons with other state-of-the-art algorithms on the ZDT problems. The results showed a competitive method, able to incorporate multiple decision maker's preferences and to perform better than some established algorithms such as NSGA-II, C-NSGA-II and SPEA2, with higher values for the hypervolume indicator and more robustness. Even though MOMCEDA is outperformed by the SMS-EMOA versions on these metrics in some cases, the performance trade-off of our technique can be verified when comparing these algorithms with the IGD metric. In this research, MOMCEDA's performance was evaluated only for bi-dimensional MOO problems, which does not imply that the algorithm cannot be extended to solve problems with more than two objective functions. However, given the fact that our method is based on an *a priori* approach, previous knowledge about the problem may be useful to help the user make proper choices of the criteria set, the relative relevance priorities and the parameter settings, in order to improve the algorithm's performance.

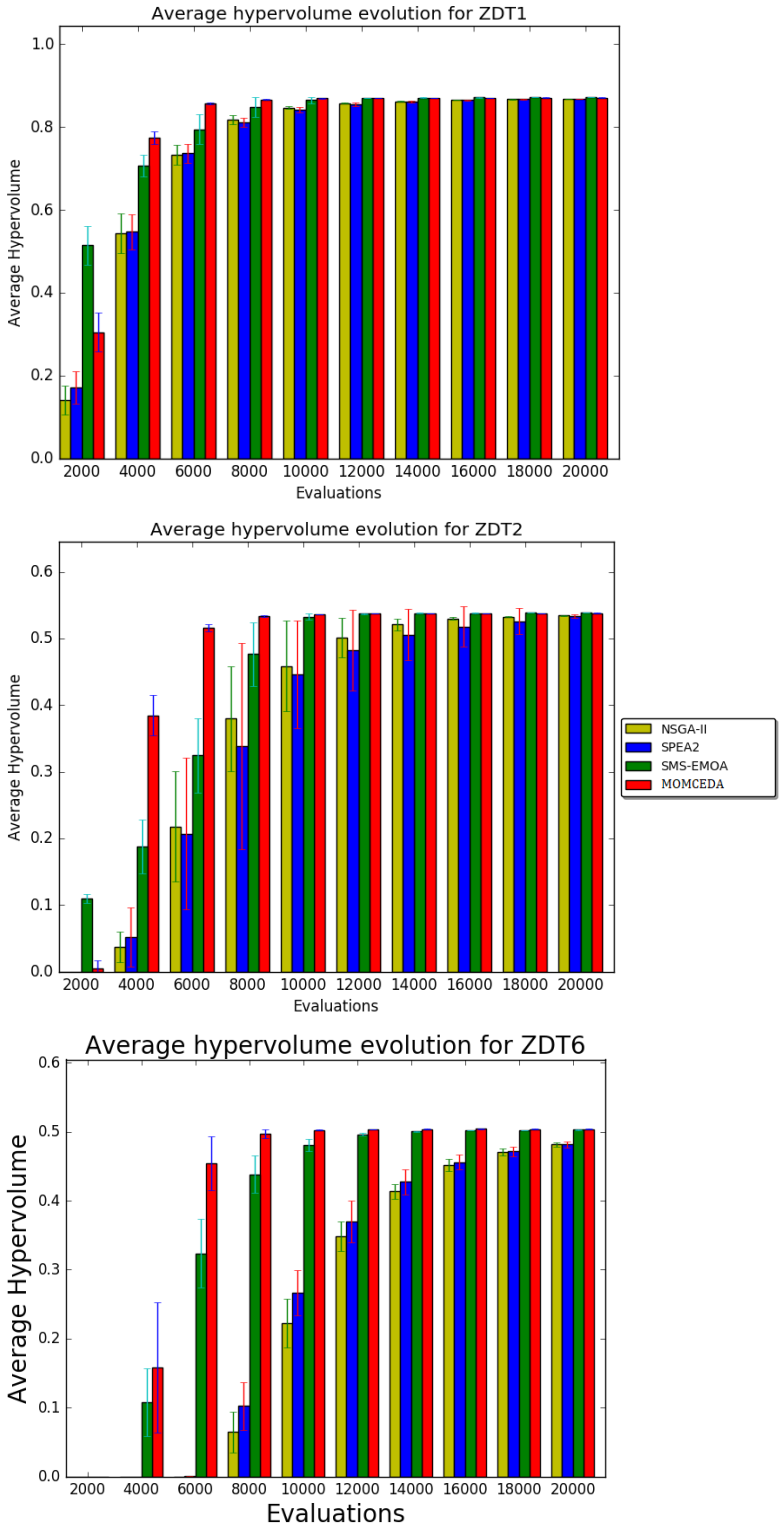

Figure 12 – Average hypervolume evolution comparison for ZDT1, ZDT2 and ZDT6 problems.

Table 7 – Hypervolume and convergence metric results.

Test function	Algorithm	Convergence Measure			Hypervolume		
		Average	Std. dev.	Rank	Average	Std. dev.	Rank
	NSGA-II	0.00054898	6.62e-05	5	0.8701	3.85e-04	9
	C-NSGA-II	0.00061173	7.86e-05	6	0.8713	2.25e-04	4
	NSGA-III	0.00098752	5.37e-05	8	0.8703	8.53e-05	6
	SPEA2	0.00100589	12.06e-05	9	0.8708	1.86e-04	5
ZDT1	ϵ-MOEA	**0.00039545**	1.22e-05	1	0.8702	8.25e-05	8
$n = 30$	MOEA/D	0.00081172	10.94-05	7	0.8706	2.40e-04	7
	SMS-EMOA	0.00044394	2.88e-05	2	**0.8721**	2.26e-05	2
	SMS-EMOA mdrp	0.00048308	2.77e-05	4	**0.8721**	7.54e-06	1
	MOMCEDA	0.00046290	7.90e-05	3	0.8713	7.46e-05	3
	NSGA-II	**0.00037851**	1.88e-05	1	0.5372	3.01e-04	8
	C-NSGA-II	0.00040011	1.91e-05	3	0.5374	4.42e-04	7
	NSGA-III	0.00111081	1.48e-04	9	0.5366	4.72e-04	9
	SPEA2	0.00082852	11.38e-05	8	0.5374	2.61e-04	6
ZDT2	ϵ-MOEA	0.00046448	2.47e-05	6	0.5383	6.39e-05	3
$n = 30$	MOEA/D	0.00063619	14.97e-05	7	0.5378	3.91e-04	5
	SMS-EMOA	0.00041004	2.34e-05	4	0.5388	3.60e-05	2
	SMS-EMOA mdrp	0.00038116	1.28e-05	2	**0.5389**	8.94e-06	1
	MOMCEDA	0.00043439	4.28e-05	5	0.5382	4.28e-05	4
	NSGA-II	0.00232321	13.95e-05	7	1.3285	1.72e-04	5
	C-NSGA-II	0.00239445	12.30e-05	8	1.3277	9.82e-04	6
	NSGA-III	0.00100283	8.61e-05	4	1.3253	5.76e-04	8
	SPEA2	0.00260542	15.46e-05	9	1.3276	2.54e-04	7
ZDT3	ϵ-MOEA	0.00175135	7.45e-05	6	1.3287	1.31e-04	3
$n = 30$	MOEA/D	0.00067269	4.99e-05	3	1.3252	2.64e-04	9
	SMS-EMOA	0.00057233	5.81e-05	2	1.3295	2.11e-05	2
	SMS-EMOA mdrp	**0.00054332**	1.83e-05	1	**1.3296**	7.49e-06	1
	MOMCEDA	0.00123011	2.33e-04	5	1.3286	1.51e-04	4
	NSGA-II	0.00639002	0.0043	8	0.8613	0.00640	5
	C-NSGA-II	0.00618386	0.0744	7	0.8558	0.00301	7
	NSGA-III	0.00377216	0.0017	5	0.8436	0.01742	9
	SPEA2	0.00769278	0.0043	9	0.8609	0.00536	6
ZDT4	ϵ-MOEA	0.00259063	0.0006	4	0.8509	0.01537	8
$n = 10$	MOEA/D	0.00241215	0.0012	2	0.8679	0.0018	2
	SMS-EMOA	0.00251878	0.0014	3	0.8677	0.00258	3
	SMS-EMOA mdrp	**0.00237157**	0.0005	1	**0.8684**	0.00094	1
	MOMCEDA	0.00428322	0.0038	6	0.8652	0.0038	4
	NSGA-II	0.07896111	0.0067	8	0.3959	0.00894	9
	C-NSGA-II	0.07940667	0.0110	9	0.3990	0.01154	8
	NSGA-III	0.01234263	0.0012	6	0.4668	0.00168	5
	SPEA2	0.00573584	0.0009	4	0.4968	0.00117	4
ZDT6	ϵ-MOEA	0.06792800	0.0118	7	0.4112	0.01573	7
$n = 10$	MOEA/D	**0.00030160**	9.60e-06	1	**0.5045**	1.73e-06	1
	SMS-EMOA	0.05043192	0.0217	5	0.4354	0.02957	6
	SMS-EMOA mdrp	0.00055733	0.00003	2	0.5040	0.00005	3
	MOMCEDA	0.00817433	0.00019	3	0.5041	0.00019	2

Table 8 – IGD metric results.

Test function	Algorithm	IGD metric	
		Average	Std. dev.
ZDT1 $n = 30$	NSGA-II	0.005212	2.47e-04
	SPEA2	0.004570	2.10e-04
	SMS-EMOA	0.003738	3.00e-04
	MOMCEDA	**0.001574**	8.85e-05
ZDT2 $n = 30$	NSGA-II	0.011373	4.04e-03
	SPEA2	0.010857	3.93e-03
	SMS-EMOA	0.010375	3.79e-03
	MOMCEDA	**0.001222**	1.51e-04
ZDT4 $n = 10$	NSGA-II	0.009392	3.21e-03
	SPEA2	0.023939	1.90e-02
	SMS-EMOA	0.012299	3.92e-03
	MOMCEDA	**0.005738**	1.44e-03
ZDT6 $n = 10$	NSGA-II	0.020354	2.65e-03
	SPEA2	0.020700	3.95e-03
	SMS-EMOA	**0.004001**	2.36e-04
	MOMCEDA	0.066887	2.05e-02

7 Conclusion

The main goal of this research was to contribute to the insertion of MCDM techniques to promote iterative decision making in MOO population based meta-heuristics, improving its computational performance toward better solutions. As result of the study developed in this work, we proposed a hybrid technique, denoted by MOMCEDA (Multi-Objective Multi-Criteria Estimation of Distribution Algorithm), composed of MOO and MCDM methods in an *a priori* approach. MOO problems arise in relevant research and application areas, and evolutionary meta-heuristics have proven to be powerful and flexible search methodologies that have successfully tackled practical challenging problems, being able to produce good-quality solutions in reasonable computation times and good enough for practical purposes.

The first chapters of this dissertation focused on presenting the three research areas concerned by this work: Multi-Objective Optimization, Multi-Criteria Decision Making and Estimation of Distribution Algorithms. In Chapter 2, we introduced single and multi-objective optimization problems, defined important concepts such as the domination among solutions and Pareto optimality, and presented some important EMOA's to solve MOO problems that were used as inspiration to conceive our method: NSGA-II, NSGA-III and SMS-EMOA. We saw that most state-of-the-art algorithms sort their candidate solution according to only one criterion at a time: first, the non-domination level is considered, and in case of a tie, the diversity in the objective space or the contribution to a performance metric is evaluated. This is the first distinction between these classical algorithms and our method, since MOMCEDA is able to classify solutions simultaneously according to multiple criteria with the help of a MCDM technique.

Chapter 3 was devoted to introduce Multi-Criteria Decision Making problems, emphasizing its applications to support iterative decision making processes in EMOA to solve MOO problems. We defined the classification of hybrid compositions of EMOA and MCDM techniques according to when the preference information is incorporated: before, after or during the search process - *a priori*, *a posteriori* and *interactive* approaches, respectively, each one having its advantages and disadvantages. The TOPSIS algorithm, the MCDM technique adopted in MOMCEDA to classify the candidate solutions, was presented in this chapter. It is considered an *a priori* approach, since the decision maker needs to define the relative relevance of the criteria before the search process, which implies that the user needs to have a good previous knowledge of the problem to define his/her preferences properly.

In Chapter 4, we discussed Estimation of Distribution Algorithms (EDA's), a class of

evolutionary optimization methods where new candidate solutions are generated by building and sampling explicit probabilistic models of promising candidate solutions among the already found ones. When compared to traditional optimization techniques, an improvement on the performance and robustness of the search for solutions is observed on EDA's. MOMCEDA makes use of a distribution model, more specifically a Gaussian mixture model, to represent its candidate solutions. This model is formed by a composition of Gaussian functions, each representing a member of the population in the decision space. The model parameters are obtained from the information provided by the TOPSIS algorithm. Since distribution models are based on the acquired experience throughout the search, unexplored promising regions of the search space may not be represented by them. Another important contribution of this research is the special sampling scheme adopted to generate new members, which was conceived to overcome this difficulty.

In Chapter 5, MOMCEDA was formally introduced. The main structure of the evolutionary meta-heuristic follows the one present in NSGA-II and NSGA-III, but endowed with distinct sorting and sampling mechanisms. The classification of candidate solutions is performed by the TOPSIS algorithm, based on user preferences derived from state-of-the-art decision policies already proposed as part of well-established meta-heuristics: the non-domination level, the neighborhood size, the distance to the closest reference direction and the exclusive contribution to the hypervolume indicator. The process of sampling new candidate solutions is implemented with the aid of a mixture of Gaussians, so that regions in the search space containing higher quality candidate solutions tend to be explored more intensively. The special sampling scheme adopted by MOMCEDA generates new candidate solutions in pairs and makes use of refreshing operators. The standard deviation of the Gaussian functions is defined in terms of the distance of the individuals in the decision space, motivated by the properties of the successful SBX operator, widely adopted in real-coded Genetic Algorithms.

The final chapter summarized the computational experiments performed to evaluate MOMCEDA's performance. The chapter began by presenting the adopted test problems and performance metrics. Then, an analysis of the Gaussian mixture model was carried out, by visualizing its probability distribution function at three different moments of the search for the Two-On-One problem. The results revealed an expected behavior: the model evolved to set higher probabilities for individuals located close to the Pareto set, which was known *a priori*. In the following sections, a series of tests was executed to evaluate the impact on the performance of the choices of different aspects of the algorithm: the type of weighting coefficients function, the priorities for the relative relevance of the user preferences and the refreshing operators. With respect to the weighting coefficients function, no alternative was

significantly superior to the others in all cases, but the exponential and linear decreasing functions had more consistent results. Regarding the relative relevance of the user preferences, the priority order that achieved the best results had the non-domination level as the highest priority criterion, followed by the diversity criteria, and finishing with the exclusive contribution to the hypervolume indicator with the lowest priority. As for the refreshing operators, the results of the tests reassured their importance to help the algorithm overcome the aforementioned difficulty.

Finally, MOMCEDA's performance was compared to several state-of-the-art algorithms on the classical ZDT test problems. The hypervolume evolution comparison revealed that our method converged faster towards the Pareto front in the first generations, but was surpassed on later generations by SMS-EMOA on the ZDT1 and ZDT2 problems, implicating that there is room for improvement of solutions at this stage of the search. The comparison of final values of hypervolume indicator placed MOMCEDA in a competitive position: it performed better than some established algorithms such as NSGA-II, C-NSGA-II and SPEA2, with higher values for the indicator and more consistent results. Regarding the convergence measure, our method did not perform so well, probably due to the limitations imposed by the NSGA-III criteria adopted by MOMCEDA, which tends to give better rankings to the solutions closer to the reference directions. When the IDG metric was evaluated, however, MOMCEDA outperformed the other algorithms in most of the cases.

Regarding future studies, we propose further investigations to analyze MOMCEDA's behavior on later generations, in order to improve the quality of the candidate solutions at this stage. The refreshing operators seem to have a great impact on the search performance, and can be a starting point of these investigations. As for the MCDM technique, another suggestion is to evaluate the impact of using other sets of criteria and other alternatives to define and normalize the values for the array of relative relevance of the user preferences (HUANG, 2008). Finally, MOMCEDA can also be extended to be tested on other benchmark problems, with three objectives and more, and also on real-world problems. Given the algorithm's nature, previous knowledge about the problem may be useful to help the user properly set his/her preferences, the relative relevance priorities and the parameter settings, aiming at improving its performance on different scenarios.

Bibliography

BEHZADIAN, M.; OTAGHSARA, S.; YAZDANI, M.; IGNATIUS, J. A state-of the-art survey of TOPSIS applications. *Expert Systems with Applications*, v. 39, n. 17, p. 13051–13069, 2012. Cited on page 37.

BEUME, N.; NAUJOKS, B.; EMMERICH, M. SMS-EMOA: Multiobjective selection based on dominated hypervolume. *European Journal of Operational Research*, v. 181, n. 3, p. 1653–1669, 2007. Cited 4 times on pages 15, 29, 53, and 67.

BISHOP, C. *Pattern Recognition and Machine Learning (Information Science and Statistics*. 1. ed. [S.l.]: Springer, 2007. ISBN 978-0-387-31073-2. Cited on page 42.

BOSMAN, P.; THIERENS, D. Mixed ideas. *Technical Report UU-CS-2000-45*, Utrech University, 2000. Cited on page 42.

BRANKE, J.; DEB, K.; MIETTINEN, K.; SLOWIńSKI, R. *Multiobjective Optimization: Interactive and Evolutionary Approaches*. [S.l.]: Springer, 2008. ISBN 978-3-540-88908-3. Cited on page 14.

COELHO, G. P. Redes Imunológicas Artificiais para Otimização em Espaços Contínuos: Uma Proposta Baseada em Concentração de Anticorpos. Universidade Estadual de Campinas - Unicamp, PhD thesis, p. 260, 2011. Cited on page 56.

COELLO COELLO, C.; LAMONT, G. B.; VELDHUIZEN, D. A. V. *Evolutionary Algorithms for Solving Multi-Objective Problems*. [S.l.]: Springer, 2007. Cited 2 times on pages 15 and 25.

DEB, K.; AGRAWAL, R. B. Simulated Binary Crossover for Continuous Search Space. *Complex Systems*, v. 9, p. 1–34, 1994. Cited 2 times on pages 17 and 50.

DEB, K.; JAIN, H. An Evolutionary Many-Objective Optimization Algorithm Using Reference-point Based Non-dominated Sorting Approach, Part I: Solving Problems with Box Constraints. *IEEE Transactions on Evolutionary Computation*, v. 18, n. 4, p. 577–601, 2013. ISSN 1089-778X. Cited 5 times on pages 15, 28, 53, 59, and 67.

DEB, K.; KUMAR, A. Interactive evolutionary multi-objective optimization and decision-making using reference direction method. *GECCO 2007: Proceedings of the Genetic and Evolutionary Computation Conference*, ACM, London, England,UK, v. 1, 2007. Cited on page 36.

DEB, K.; KUMAR, A. Light beam search based multi-objective optimization using evolutionary algorithms. *2007 IEEE Congress on Evolutionary Computation (CEC)*, IEEE, p. 2125–2132, 2007. Cited on page 36.

DEB, K.; MOHAN, M.; MISHRA, S. Towards a Quick Computation of Well-Spread Pareto-Optimal Solutions. In: FONSECA, C. M.; FLEMING, P. J.; ZITZLER, E.; THIELE,

L.; DEB, K. (Ed.). *Evolutionary Multi-Criterion Optimization*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003. p. 222–236. ISBN 978-3-540-36970-7. Cited 3 times on pages 36, 58, and 67.

DEB, K.; PRATAP, A.; AGARWAL, S.; MEYARIVAN, T. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, v. 6, n. 2, p. 182–197, 2002. Cited 5 times on pages 14, 15, 22, 25, and 26.

DEB, K.; SUNDAR, J. Reference point based multi-objective optimization using evolutionary algorithms. *GECCO 2006: Proceedings of the Genetic and Evolutionary Computation Conference*, ACM, New York, New York, USA, p. 635–642, 2006. Cited on page 35.

DENNIS, J.; DAS, I. Normal-boundary intersection: a new method for generating pareto optimal points in nonlinear multicriteria optimization problems. *SIAM J Optim*, v. 8, n. 3, p. 631–657, 1998. Cited on page 28.

DYER, J.; FISHBURN, P.; STEUER, R.; WALLENIUS, J.; ZIONTS, S. Multiple criteria decision making, multiattribute utility theory: The next ten years. *Management Science*, v. 38, n. 5, p. 645–654, 1992. Cited on page 34.

FONSECA, C.; FLEMING, P. Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. *Proceedings of the 5th International Conference on Genetic Algorithms*, Morgan Kaufmann Publishers Inc., p. 416–423, 1993. Cited on page 35.

FORTIN, F.-A.; De Rainville, F.-M.; GARDNER, M.-A.; PARIZEAU, M.; GAGNÉ, C. DEAP: Evolutionary algorithms made easy. *Journal of Machine Learning Research*, v. 13, p. 2171–2175, jul 2012. Cited on page 67.

GONÇALVES, A. R. Otimização em ambientes dinâmicos com variáveis contínuas empregando algoritmos de estimação de distribuição. Universidade Estadual de Campinas - Unicamp, MEng thesis, p. 25–29, 2011. Cited on page 41.

GOULART, F. Preference-guided Evolutionary Algorithms for Optimization with Many Objectives. Universidade Federal de Minas Gerais - UFMG, MEng thesis, p. 65–68, 2014. Cited on page 35.

HANDA, H. The effectiveness of mutation operation in the case of estimation of distribution algorithms. *Biosystems*, v. 87, n. 2, p. 243 – 251, 2007. Papers presented at the Sixth International Workshop on Information Processing in Cells and Tissues, York, UK, 2005. Cited on page 43.

HAUSCHILD, M.; PELIKAN, M. An introduction and survey of estimation of distribution algorithms. *Swarm and Evolutionary Computation*, v. 1, p. 111–128, 2011. Cited 3 times on pages 16, 41, and 43.

HUANG, J. Combining entropy weight and TOPSIS method for information system selection. In: *2008 IEEE Conference on Cybernetics and Intelligent Systems*. [S.l.: s.n.], 2008. p. 1281–1284. ISSN 2326-8123. Cited on page 75.

HWANG, C.; YOON, K. *Multiple Attributes Decision Making Methods and Applications*. Berlin: Springer-Verlag, 1981. Cited 3 times on pages 16, 35, and 37.

KÖKSALAN, M.; WALLENIUS, J.; ZIONTS, S. *Multiple Criteria Decision Making: From Early History to the 21st Century*. [S.l.]: World Scientific, 2011. Cited on page 14.

LARRAÑAGA, P.; LOZANO, J. *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation*. [S.l.]: Kluwer Academic Publishers, 2001. Cited on page 40.

LEBESGUE, H. Intégrale, Longueur, Aire. Université de Paris, PhD thesis, 1902. Cited on page 30.

MIETTINEN, K. M. *Nonlinear Multiobjective Optimization*. [S.l.]: Kluwer, 1999. ISBN 978-1-4615-5563-6. Cited on page 34.

MODIRI-DELSHAD, M.; RAHIM, N. A. Multi-objective backtracking search algorithm for economic emission dispatch problem. *Applied Soft Computing*, v. 40, p. 479–494, 2016. Cited on page 14.

MOLINA, J.; SANTANA, L. V.; HERNÁNDEZ-DÍAZ, A. G.; COELLO COELLO, C. A.; CABALLERO, R. g-dominance: Reference point based dominance for multiobjective metaheuristics. *European Journal of Operational Research*, v. 197, n. 2, p. 685–692, 2009. Cited on page 35.

NAUJOKS, B.; BEUME, N.; EMMERICH, M. Multi-objective optimisation using S-metric selection: application to three-dimensional solution spaces. *2005 IEEE Congress on Evolutionary Computation*, v. 2, p. 1282–1289, 2005. Cited on page 32.

PARZEN, E. On estimation of a probability density function and mode. *Ann. Math. Statist.*, The Institute of Mathematical Statistics, v. 33, n. 3, p. 1065–1076, 09 1962. Cited on page 42.

PREUSS, M.; NAUJOKS, B.; RUDOLPH, G. Pareto Set and EMOA Behavior for Simple Multimodal Multiobjective Functions. In: RUNARSSON, T. P.; BEYER, H.-G.; BURKE, E.; MERELO-GUERVÓS, J. J.; WHITLEY, L. D.; YAO, X. (Ed.). *Parallel Problem Solving from Nature - PPSN IX*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006. p. 513–522. ISBN 978-3-540-38991-0. Cited 4 times on pages , 55, 61, and 63.

PURSHOUSE, R. C.; DEB, K.; MANSOR, M. M.; MOSTAGHIM, S.; WANG, R. A Review of Hybrid Evolutionary Multiple Criteria Decision Making Methods. *IEEE Congress on Evolutionary Computation (CEC), Beijing, China*, p. 1147–1154, 2014. Cited 3 times on pages 16, 35, and 36.

RAO, S. S. *Engineering Optimization: Theory and Practice*. 4. ed. [S.l.]: John Wiley & Sons, Inc., 2009. ISBN 9780470183526. Cited on page 21.

ROSENBLATT, M. Remarks on some nonparametric estimates of a density function. *Ann. Math. Statist.*, The Institute of Mathematical Statistics, v. 27, n. 3, p. 832–837, 09 1956. Cited on page 42.

SAN CRISTÓBAL, J. R. *Multi Criteria Analysis in the Renewable Energy Industry*. [S.l.]: Springer, 2012. 7–11 p. ISBN 978-1-4471-2345-3. Cited on page 34.

SHIM, V. A.; TAN, K. C.; CHEONG, C. Y.; CHIA, J. Y. Enhancing the scalability of multi-objective optimization via restricted boltzmann machine-based estimation of distribution algorithm. *Information Sciences*, v. 248, p. 191 – 213, 2013. Cited on page 42.

SOUSA BEZERRA, P. M.; VON ZUBEN, F. J.; COELHO, G. P. Multi-objective optimization based on a multi-criteria estimation of distribution. *Proceedings of the 2018 International Conference on Artificial Intelligence*, CSREA Press, p. 143–149, 2018. Cited 2 times on pages 45 and 55.

VON ZUBEN, F. J.; COELHO, G. P. *Evolutionary Multi-Objective Optimization, classroom notes*. 2017. Cited 2 times on pages and 25.

WESSING, S. *evoalgos: Modular evolutionary algorithms*. 2017. [Online; accessed January 27, 2018]. Cited on page 67.

WIERZBICKI, A. P. A mathematical basis for satisficing decision making. *Mathematical Modelling*, v. 3, n. 5, p. 391–405, 1982. Cited 2 times on pages 15 and 24.

WRIGHT, S. J. *Optimization*. [S.l.]: Encyclopædia Britannica, inc., 2016. [Online; accessed August 30, 2018]. Cited on page 19.

YANG, Z.; EMMERICH, M.; BäCK, T.; KOK, J. Multicriteria inventory routing by cooperative swarms and evolutionary algorithms. *Bioin- spired Computation in Artificial Systems*, Springer, p. 127–137, 2015. Cited on page 14.

YANG, Z.; WANG, H.; YANG, K.; BACK, T.; EMMERICH, M. SMS-EMOA with multiple dynamic reference points. *2016 12th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery, ICNC-FSKD 2016*, p. 282–288, 2016. Cited 2 times on pages 67 and 68.

YAO, G.; DING, Y.; JIN, Y.; HAO, K. Endocrine-based coevolutionary multi-swarm for multi-objective workflow scheduling in a cloud system. *Soft Computing*, p. 1–14, 2016. Cited on page 14.

ZHANG, Q.; LI, H. MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition. *IEEE Transactions on Evolutionary Computation*, v. 11, n. 6, p. 712–731, 2007. Cited on page 36.

ZITZLER, E.; DEB, K.; THIELE, L. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation*, v. 8, p. 173–195, 2000. Cited on page 55.

ZITZLER, E.; LAUMANNS, M.; THIELE, L. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. *Evolutionary Methods for Design Optimization and Control with Applications to Industrial Problems*, p. 95–100, 2001. Cited on page 15.

ZITZLER, E.; THIELE, L. Multiobjective optimization using evolutionary algorithms — A comparative case study. *Parallel Problem Solving from Nature - PPSN V*, p. 292–301, 1998. Cited 2 times on pages 15 and 30.