UNIVERSITÀ
DEGLI STUDI
FIRENZE

Università degli Studi di Firenze
Dipartimento di Ingegneria dell'Informazione (DINFO)
Corso di Dottorato in Ingegneria dell'Informazione
Curriculum: Automatica, Ottimizzazione e Sistemi Complessi

Pareto Front Reconstruction of Multi-Objective Optimization Problems

Candidate
Pierluigi Mansueto

Supervisors
Prof. Fabio Schoen

Prof. Marco Sciandrone

PhD Coordinator
Prof. Fabio Schoen

ciclo XXXVI, 2020-2023

Università degli Studi di Firenze, Dipartimento di Ingegneria
dell'Informazione (DINFO).

A Martina,
Esempio di Vita

A Mamma e Babbo,
con Me dal Principio

A Elia e Michela,
Esempi di Coraggio

A Sara,
Sempre al mio Fianco

Acknowledgments

For many days, I have thought about how I could structure this part. These three years represent an incredible journey to me: there were many "victories", academically speaking, some "failures", but, in particular, I had amazing people by my side. This part of the thesis is exactly about them: colleagues, friends, family members...without them, nothing written in this manuscript would have been possible.

I will start from my tutors, prof. Fabio Schoen and prof. Marco Sciandrone. I deeply thank you, not only for your excellent skills on operations research, but also for the great people you are beyond the academic world.

After my tutors, the stage is all for Matteo Lapucci. I am not exaggerating if I say that Matteo is the guy who contributed most to make me the person who I am, professionally speaking. But, Matteo is not only that...he is also a friend, with whom you can make a laugh during the days or have crazy return trips. I hope that the life always bless you, Matte!

Now, it is the turn of my "colleagues", people that make the GOL laboratory a beautiful and stimulating working place. I am talking about Enrico, Simone, Alessio, Tommaso, Francesco, Leonardo, Tomaso and Marco. Spending the days with you was an invaluable gift, and I am not talking only from a professional perspective.

I want to mention my friends, fundamental part of my everyday life. We had great moments and many lifetime memories which I will jealously take care of. And I hope that, in the future, there will be many more. Andrea Giuntini, my friend, you had a special place in this context. I will never get tired of saying how much a "reference point" you are for me...it is not important how far away we are from each other: I will always be there for you.

Sara, it is your turn. Maybe, this part was the most difficult to write, because telling how much I care of you is not enough. You were always there for me, not only in the best moments (this was easy), but even in the most sad or difficult ones of my life, without having second thoughts. Just this last sentence should indicate the great woman you are. I will return the favor, hopefully for a long time.

Thanks to my sisters and brother: Michela, Martina and Elia. You are the greatest people I know, so much different, yet so close. You are the proof that, in this life, determination, backbone and courage must never be missing.

A special mention goes to grandpa Fulvio, grandma Vera, aunt Roberta, uncle Filippo and my cousins, Gaia, Francesco and Alessandro: they took care of me from the beginning with affection and love.

Last but not least, this thesis is dedicated to my parents, Armando and Luana: the man I am is because of you and, for this reason, I love you more every day.

Abstract

The thesis is concerned with multi-objective optimization (MOO) problems under various constraint types. In particular, we consider the unconstrained, the box constrained, the general convex constrained and the cardinality constrained settings. As mathematical tool, MOO has received much attention over the years, being suitable both in operation research applications and in many real-world problems where contrasting goals have to be taken into account.

In the dissertation, a general overview of the main MOO concepts is given, focusing on the notions typically employed in the gradient-based MOO approaches. In particular, we present a general formulation for the search direction problem and a general framework for single-point methodologies, i.e., approaches designed to generate a single solution to the problem at hand. The features of each of the two concepts are discussed and analyzed; finally, we show how they can be reduced to well-know schemes from the MOO literature.

Then, we propose novel gradient-based methodologies aimed to reconstruct the Pareto front for the considered problem classes. In some of the mentioned settings it is the first attempt to define this type of approaches generating multiple solutions rather than one, while in the others the topic has been almost unexplored yet. However, in the MOO context, returning an approximation of the Pareto front rather than a single solution can be much more useful for the final user, so as they can choose among multiple trade-offs of the objectives the one that is the most suitable for themselves. For each methodology, detailed descriptions of the algorithmic scheme and characteristic features are reported; moreover, each methodology is theoretically analyzed and its convergence properties are stated and proved.

Finally, the proposed methods are numerically tested with wide benchmarks of test problems, comparing each of them with state-of-the-art approaches from the MOO literature. The results show the efficiency and the effectiveness of the proposals w.r.t. the competitors in diverse experimental settings.

Contents

Chapter 1

Introduction

Multi-objective optimization (MOO) has received a lot of attention by the research community for the last 25 years. As a matter of fact, MOO problems turned out to be relevant in different application fields, where objectives that are in contrast with each other must be taken into account. Engineering [17, 88, 96], management [45, 105], statistics [19], space exploration [86, 98] are just some context examples where novel and effective multi-objective optimization applications have been considered.

The interests on studying MOO rely on two major complexities, that coupled together make MOO problems particularly difficult to handle. The first complexity element is the general absence of a solution minimizing all the objective functions simultaneously; as a consequence, the definitions of optimality (global, local and stationarity), based on Pareto's theory, are not trivial and make optimization processes not obvious, both in terms of aims and tools. The second one, on the other hand, traces one classical issue typical of scalar optimization: in absence of convexity assumptions, there is no equivalence between local and global (Pareto) optimality.

Popular classes of algorithms to solve multi-objective problems are those of scalarization methods [30, 32, 35, 42, 43, 87, 95, 106] and of heuristic methods based on genetic and evolutionary algorithms (EAs) [56, 63, 81]. Among genetic methods, the `NSGA-II` algorithm [26] is arguably the most popular; basically, it is a population-based procedure exploiting a cheaply computable score to efficiently rank solutions w.r.t. the objectives and performing the classical genetic crossover, mutation and selection operations to create the new generation of solutions. However, both the scalarization and the evo-

lutionary approaches come with non-negligible shortcomings. Indeed, the first ones require a detailed analysis of the problem structure in order to identify the weights defining a suitable scalarized objective. Moreover, an unfortunate choice of the weights may lead to unbounded scalar problems, even under strong regularity assumptions [34, sec. 7]. Finally, unlike heuristic approaches such as `NSGA-II`, scalarization is designed to produce a single solution and, in order to generate an approximation of the whole Pareto front, the problem has to be repeatedly solved with different not known a priori choices of weights. On the other hand, convergence properties cannot be stated for heuristic algorithms; moreover, they might be expensive on some scenarios [49, 66, 94].

A class of MOO methods that has been widely studied for the past two decades is the one concerning descent algorithms (either first-order, second-order and derivative-free). These approaches are basically extensions of the classical iterative scalar optimization algorithms. Steepest Descent [36], Newton [34, 44], Quasi-Newton [89], Augmented Lagrangian [21], Conjugate Gradient [71] are only a few methods of this family. In addition to having theoretically relevant convergence properties, these algorithms, when used on problems with reasonable regularity assumptions, have proven to be valid alternatives to the scalarization approaches [87] and the evolutionary ones [26], especially as the problem size grows [23]. The earliest developed algorithms of this class are *single-point*, i.e., only able to produce a single Pareto-stationary solution; in order to generate an approximation of the entire Pareto front, they must be run multiple times in a multi-start fashion. More recently, some of these algorithms have been extended to overcome this limitation. These new *front-oriented* approaches (see, e.g., [23–25, 37]) are capable of dealing with sequences of sets of points and thus producing a Pareto front approximation in an efficient and effective manner. Indeed, in the context of multi-objective applications, it is in practice crucial to generate a set of solutions constituting an approximation of the Pareto set, so that the user can choose, a posteriori, the solution providing the most appropriate trade-off among many.

In this thesis, we focus on the development of novel gradient-based approaches for multi-objective optimization problems, both unconstrained and convex/cardinality constrained. In particular, we study new single-point and front-oriented algorithms with the ultimate goal of proposing methodologies capable of reconstructing the Pareto front in the most effective and efficient

way. For each proposal, we describe the algorithmic scheme and provide a detailed theoretical analysis, where properties, including the convergence ones, are stated and proved. Finally, in order to test the efficiency and effectiveness of the new methods, they are compared with state-of-the-art algorithms for MOO on benchmarks of test problems; the results of these extensive computational experiments are shown and commented. Going more into the details, the rest of the thesis concerns the following contributions.

- In Chapter 2, we recall the main concepts of Pareto's theory; moreover, we propose a general formulation for the search direction optimization problem and we show how the latter can be reduced to well-known schemes to find standard directions.

- In Chapter 3, we review some of the standard single-point gradient-based methods, introducing a general framework that can represent each of them; furthermore, we state some properties of the considered algorithms that will be useful for the next sections; finally, we report a brief theoretical description of the *Front Steepest Descent* (`FSD`) [23] algorithm, which is designed to reconstruct the Pareto front of unconstrained multi-objective optimization problems.

- In Chapter 4, we propose a memetic algorithm for bound-constrained MOO problems that inherits the good features of both EA and MO descent families. In particular, the new approach, called *Non-dominated Sorting Memetic Algorithm* (`NSMA`), consists on combining the popular genetic approach `NSGA-II` with MO descent methods, similarly to what is done in the scalar case in [47]. In this context, we present a new front-oriented descent method, particularly suited to be executed in combination with `NSGA-II`, and we provide a theoretical analysis where feasibility and convergence properties are stated and proved.

- In Chapter 5, a multi-objective limited memory Quasi-Newton algorithm for unconstrained MOO problems is presented. The possibility of defining limited-memory variants of Quasi-Newton methods certainly stands out among the factors contributing to the success of this class of methods in scalar optimization: the management in memory of the Hessian matrix, extremely inefficient and time-consuming in many cases, is avoided but, still, an approximation of it with only the previously generated solutions is provided. To the best of our knowledge,

limit-memory approaches for MOO have not been studied in the literature yet and our proposal represents de facto the first of this class of algorithms.

- In Chapter 6, we focus on the `FSD` [23] algorithm, arguing its limited exploration capabilities which prevent it from spanning large portions of the Pareto front; we thus propose small but crucial modifications to the algorithm, showing that the new approach still preserves the same convergence guarantees.

- In Chapter 7, we propose an extended version of the augmented Lagrangian algorithm for multi-objective optimization (`ALAMO`) proposed in [21], which deals with sets of points and effectively produces an approximation of the Pareto front for convex constrained multi-objective optimization problems; unlike the original single-point `ALAMO`, this front-oriented version, called `FRONT-ALAMO`, makes use of a common penalty parameter and Lagrange multipliers for all points in the set of solutions; moreover, unlike the other approaches presented in this thesis, `FRONT-ALAMO` has properties of convergence to Pareto-stationarity that can be proved without recurring to concepts such as the linked sequence [67].

- In Chapter 8, we introduce new optimality conditions for MOO problems with cardinality constraints, whose theoretical foundation was only recently established in [57]; then, we present two new algorithms to solve these problems; the first one is a single-point method consisting on an extension of the `IHT` method [3] to the MOO case; the second algorithm, on the other hand, is a two-stage approach whose ultimate goal is to approximate the whole Pareto front; for both methods, we conduct a rigorous theoretical analysis, proving their properties of convergence to points satisfying necessary conditions for global optimality.

- In Chapter 9, we report the results of thorough computational experiments, comparing the performance of our proposals with the one of state-of-the-art algorithms from the multi-objective optimization literature on varied benchmarks of test problems.

- In Chapter 10, we provide some concluding remarks and possible ideas for future research.

Chapter 2

Preliminaries

In this chapter, we present the general form of the multi-objective optimization problems we will address in the thesis. Moreover, along with some key concepts from the Pareto's theory, we introduce a generic formulation for the search direction problem. As will be shown, this one can be easily reduced to well-known problems to find standard descent directions for multi-objective optimization.

2.1 Problem Statement

Throughout the thesis, we consider multi-objective optimization problems of the form

$$\begin{aligned} &\min F(x) = (f_1(x), \ldots, f_m(x))^\top \\ &\text{s.t. } x \in \Omega, \end{aligned} \tag{2.1}$$

where $F : \mathbb{R}^n \to \mathbb{R}^m$ is a vector-valued continuously differentiable function, i.e., $F \in C^1(\Omega, \mathbb{R}^m)$, and the feasible set Ω is assumed to be closed, non-empty and non-discrete. We denote by $J_F(\cdot) = (\nabla f_1(\cdot), \ldots, \nabla f_m(\cdot))^\top \in \mathbb{R}^{m \times n}$ the Jacobian matrix associated with $F(\cdot)$. Moreover, for all $j \in \{1, \ldots, m\}$, the Hessian matrix of the function $f_j(\cdot)$, when it exists, is denoted by $\nabla^2 f_j(\cdot)$. In what follows, we will denote the Euclidean norm in $\mathbb{R}^n$ by $\|\cdot\|$ and the N-dimensional vectors of all ones and all zeros, with $N > 0$, by $\mathbf{1}_N$ and $\mathbf{0}_N$ respectively. Finally, we denote by I_N the identity matrix of size N.

Since we are in a multi-objective setting, we introduce the standard partial ordering in $\mathbb{R}^m$: considering two points $u, v \in \mathbb{R}^m$, we have that

$$\begin{aligned} u < v &\iff u_i < v_i, \qquad \forall i = 1, \dots, m, \\ u \leq v &\iff u_i \leq v_i, \qquad \forall i = 1, \dots, m. \end{aligned}$$

An analogous definition can be also stated for the operators $>, \geq$. Furthermore, given the objective function $F(\cdot)$, we say that $x \in \Omega$ *dominates* $y \in \Omega$ w.r.t. F if $F(x) \leq F(y)$ and $F(x) \neq F(y)$ and we denote it by $F(x) \lneqq F(y)$.

In multi-objective optimization problems, we ideally would like to obtain a point which simultaneously minimizes all the objectives at once. However, such a solution is unlikely to exist. In this scenario, the concepts of optimality are based on Pareto's theory.

Definition 2.1. A point $\bar{x} \in \Omega$ is *Pareto optimal* for Problem (2.1) if a point $y \in \Omega$ such that $F(y) \lneqq F(\bar{x})$ does not exist. If there exists a neighborhood $\mathcal{N}(\bar{x})$ such that the previous property is satisfied in $\Omega \cap \mathcal{N}(\bar{x})$, then $\bar{x}$ is *locally Pareto optimal*.

In practice, it is difficult to attain solutions satisfying the Pareto optimality property. A slightly weaker but certainly more viable to obtain condition is weak Pareto optimality.

Definition 2.2. A point $\bar{x} \in \Omega$ is *weakly Pareto optimal* for Problem (2.1) if a point $y \in \Omega$ such that $F(y) < F(\bar{x})$ does not exist. If there exists a neighborhood $\mathcal{N}(\bar{x})$ such that the previous property is satisfied in $\Omega \cap \mathcal{N}(\bar{x})$, then $\bar{x}$ is *locally weakly Pareto optimal*.

We define the *Pareto set* as the set of all the Pareto optimal solutions. Moreover, we refer to the image of the Pareto set w.r.t. F as the *Pareto front*.

We can now introduce the concept of Pareto stationarity.

Definition 2.3. A point $\bar{x} \in \Omega$ is *Pareto-stationary* for Problem (2.1) if we have that

$$\max_{j=1,\dots,m} \nabla f_j(\bar{x})^\top d \geq 0, \quad \forall d \in \mathcal{D}(\bar{x}),$$

where

$$\mathcal{D}(\bar{x}) = \{v \in \mathbb{R}^n \mid \exists \bar{t} > 0 : \bar{x} + tv \in \Omega \; \forall t \in [0, \bar{t}]\}. \tag{2.2}$$

The property can be also compactly re-written as

$$\min_{d\in\mathcal{D}(\bar{x})}\max_{j=1,\dots,m}\nabla f_j(\bar{x})^\top d = 0.$$

If a point $\bar{x}$ is not a Pareto-stationary point for Problem (2.1), then there exists a direction $\hat{d}\in\mathcal{D}(\bar{x})$ such that $\max_{j=1,\dots,m}\nabla f_j(\bar{x})^\top\hat{d}<0$. Since $F(\cdot)$ is continuously differentiable, we have that

$$\lim_{t\to 0}\frac{f_j(\bar{x}+t\hat{d})-f_j(\bar{x})}{t} = \nabla f_j(\bar{x})^\top\hat{d} < 0, \qquad \forall j=1,\dots,m.$$

Thus, $\hat{d}$ is a *common descent direction* for $F(\cdot)$ at $\bar{x}$, i.e., there exists $\bar{t}>0$ such that

$$F(\bar{x}+t\hat{d}) < F(\bar{x}), \qquad \forall t\in(0,\bar{t}\,].$$

Under differentiability assumptions, Pareto stationarity is necessary for all types of Pareto optimality. Moreover, assuming the convexity of both $F(\cdot)$ (component-wise) and Ω, Pareto stationarity is also a sufficient condition for weak Pareto optimality. These theoretical relationships are stated in the next lemma.

Lemma 2.1 ([34, Theorem 3.1])**.** *The following statements hold:*

- *if $\bar{x}$ is locally weakly Pareto optimal, then $\bar{x}$ is Pareto-stationary for Problem* (2.1)*;*
- *if Ω is convex, $F(\cdot)$ is component-wise convex and $\bar{x}$ is Pareto-stationary for Problem* (2.1)*, then $\bar{x}$ is weakly Pareto optimal;*
- *if Ω is convex, $F(\cdot)$ is twice continuously differentiable, $\nabla^2 f_j(x)\succ 0$ for all $j\in\{1,\dots,m\}$ and all $x\in\Omega$, and $\bar{x}$ is Pareto-stationary for Problem* (2.1)*, then $\bar{x}$ is Pareto optimal.*

2.2 A Generic Formulation for the Search Direction Problem

In this section, we introduce a generic formulation for the search direction problem:

$$\min_{d\in\mathcal{D}(\bar{x})}\max_{j=1,\dots,m}\nabla f_j(\bar{x})^\top d + \frac{1}{2}d^\top M_j(\bar{x})d, \tag{2.3}$$

where, for all $j \in \{1, \dots, m\}$, $M_j(\bar{x}) \in \mathbb{R}^{n \times n}$. If the matrices $M_j(\cdot)$ are positive definite, i.e., $M_j(x) \succ 0 \ \forall j \in \{1, \dots, m\}$ and $\forall x \in \Omega$, the function $\nabla f_j(\bar{x})^\top d + (1/2)\, d^\top M_j(\bar{x}) d$ is strongly convex for each $j \in \{1, \dots, m\}$ and, thus, Problem (2.3) has a unique minimizer. We denote the latter by $v(\bar{x})$ and we indicate with $\theta(\bar{x})$ the optimal value of the problem at $\bar{x}$. Problem (2.3) can be also reformulated as

$$\begin{aligned} \min_{\substack{\beta \in \mathbb{R} \\ d \in \mathcal{D}(\bar{x})}} \quad & \beta \\ \text{s.t. } & \nabla f_j(\bar{x})^\top d + \frac{1}{2} d^\top M_j(\bar{x}) d \leq \beta, \quad \forall\, j \in \{1, \dots, m\}. \end{aligned}$$

Lemma 2.2. *Let us assume that, for all $j \in \{1, \dots, m\}$, the matrix $M_j(\cdot)$ is positive definite, i.e., $M_j(x) \succ 0 \ \forall x \in \Omega$. Then, for all $x \in \Omega$, we have:*

1. *$\theta(x)$ and $v(x)$ are well-defined;*
2. *$\theta(x) \leq 0$;*
3. *the following conditions are equivalent:*
   - *x is not Pareto stationary;*
   - *$\theta(x) < 0$;*
   - *$v(x) \neq \mathbf{0}_n$.*

Proof.

1. The proof is trivial since
   - the feasible set $\mathcal{D}(x)$ (Equation 2.2) is closed and non-empty,
   - $\max_{j=1,\dots,m} \nabla f_j(x)^\top d + \frac{1}{2} d^\top M_j(x) d$ is strongly convex ($M_j(x) \succ 0$ for all $j \in \{1, \dots, m\}$ and $x \in \Omega$) and continuous in $\mathcal{D}(x)$.
2. Given $\hat{d} = \mathbf{0}_n$, we have that $\max_{j=1,\dots,m} \nabla f_j(x)^\top \hat{d} + \frac{1}{2} \hat{d}^\top M_j(x) \hat{d} = 0$. Since $\hat{d} \in \mathcal{D}(x)$, we get the thesis.
3. The proof is identical to the one for [34, Lemma 3.2, Point 2.]. We can indeed replace each Hessian matrix $\nabla^2 f_j(\cdot)$, $j \in \{1, \dots, m\}$, with the matrix $M_j(\cdot)$, which is assumed to be positive definite.

□

Given the generic formulation for the search direction problem, we are ready to introduce a relaxation of Pareto stationarity, called ε-Pareto-stationarity. This concept is firstly introduced in [21]: here, we propose a slightly modified version.

Definition 2.4. Let $\varepsilon > 0$. A point $\bar{x} \in \Omega$ is ε-Pareto-stationary for Problem (2.1) if

$$\min_{d \in \mathcal{D}(\bar{x})} \max_{j=1,\dots,m} \nabla f_j(\bar{x})^\top d + \frac{1}{2}\|d\|^2 \geq -\varepsilon.$$

Note that in this last problem $M_j(\cdot) = I_n$ for all $j = 1, \dots, m$. Moreover, as in [90], we introduce the function $D : \Omega \times \mathbb{R}^n \to \mathbb{R}$, defined by

$$D(x, d) = \max_{j=1,\dots,m} \nabla f_j(x)^\top d. \tag{2.4}$$

It is trivial to see that any direction d such that $D(\bar{x}, d) < 0$ is a *common descent direction* at $\bar{x}$ for $F(\cdot)$. Moreover, the function $D(\cdot, \cdot)$ has some properties, which we report in the next lemma.

Lemma 2.3 ([90, Lemma 2.2])**.** *The following statements hold:*

- *for any $x \in \Omega$ and $\alpha \geq 0$, we have $D(x, \alpha d) = \alpha D(x, d)$.*
- *the mapping $(x, d) \to D(x, d)$ is continuous.*

Depending on the definition of the feasible set Ω and the matrices $M_1(\cdot)$, $\dots$, $M_m(\cdot)$, Problem (2.3) can be reduced to well-known schemes to define the search direction. In Table 2.1, we summarize some of the most important ones. Note that only the Newton [34] and Quasi-Newton [89] directions employ matrices different for each objective function. In particular, in the remainder of the thesis, the matrices $B_1, \dots, B_m$ will be called *approximation matrices*, as in the Quasi-Newton methodologies they aim to estimate the objective functions curvature information typically contained in the second derivatives $\nabla^2 f_1(\cdot), \dots, \nabla^2 f_m(\cdot)$. Further information about them can be found in Sections 2.2.1 and 2.2.2.

In the next lemma, we report the continuity properties related to some of the functions $v(\cdot)$, $\theta(\cdot)$ listed in the table.

Lemma 2.4. *The following statements hold:*

1. *[36, Lemma 1, Point 3.] the mappings $x \to v^{SD}(x)$ and $x \to \theta^{SD}(x)$, with $x \in \mathbb{R}^n$, are continuous;*
2. *[29, Proposition 4] the function $\theta^{CS}(\cdot)$ is continuous in C, with C being a convex set;*
3. *[34, Lemma 3.2, Point 3.] the mapping $x \to v^N(x)$, with $x \in U \subset \mathbb{R}^n$ and U being an open set, is bounded on compact sets and the function $\theta^N(\cdot)$ is continuous in U;*

Name	**Ω**	**$\mathbf{M_j}(\cdot)$ $(\mathbf{j = 1, \dots, m})$**	**Notation**
Steepest common descent direction [36]	$\mathbb{R}^n$	I_n	$v^{SD}(\cdot)$, $\theta^{SD}(\cdot)$
Constrained steepest common descent direction [29]	$C \subset \mathbb{R}^n$ (Convex)	I_n	$v^{CS}(\cdot)$, $\theta^{CS}(\cdot)$
Newton direction [34]	$\mathbb{R}^n$	$\nabla^2 f_j(\cdot)$ $(F \in C^2(\Omega, \mathbb{R}^m))$ $(\nabla^2 f_j(x) \succ 0, \forall x \in \mathbb{R}^n)$	$v^{N}(\cdot)$, $\theta^{N}(\cdot)$
Quasi-Newton direction [89]	$\mathbb{R}^n$	$B_j \succ 0$	$v^{QN}(\cdot)$, $\theta^{QN}(\cdot)$
Modified Quasi-Newton direction [1]	$\mathbb{R}^n$	$B \succ 0$	$v^{MQN}(\cdot)$, $\theta^{MQN}(\cdot)$

Table 2.1: Well-known search directions.

4. *[89, Lemma 2, Point (c)] the function $\theta_{QN}(\cdot)$ is continuous in U, with $U \subseteq \mathbb{R}^n$ being an open set.*

In the special cases where $\Omega = \mathbb{R}^n$, the dual form of Problem (2.3) can be explicitly formulated. In particular, we conducted a thorough analysis on the problem, which is reported in Appendix B, and we get that the Lagrangian dual problem of (2.3) is given by:

$$\begin{aligned} \max_{\lambda \in \mathbb{R}^m} \; & -\frac{1}{2}\lambda^\top J_F(\bar{x}) \left[\sum_{j=1}^{m} \lambda_j M_j(\bar{x})\right]^{-1} J_F(\bar{x})^\top \lambda \\ \text{s.t.} \; & \sum_{j=1}^{m} \lambda_j = 1, \qquad \lambda \geq \mathbf{0}_m. \end{aligned} \tag{2.5}$$

In this scenario, if the matrices $M_1(\cdot), \dots, M_m(\cdot)$ are positive definite, strong duality holds and the Karush-Kuhn-Tucker (KKT) conditions are sufficient and necessary for optimality. Moreover, denoting by $\lambda(\bar{x}) = (\lambda_1(\bar{x}), \dots, \lambda_m(\bar{x}))^\top$ the optimal Lagrange multipliers vector, we have that

$$\sum_{j=1}^{m} \lambda_j(\bar{x}) = 1, \qquad \lambda(\bar{x}) \geq \mathbf{0}_m \tag{2.6}$$

and

$$v(\bar{x}) = -\left[\sum_{j=1}^{m} \lambda_j(\bar{x}) M_j(\bar{x})\right]^{-1} J_F(\bar{x})^\top \lambda(\bar{x}). \tag{2.7}$$

In the remainder of the thesis, we will use for the Lagrange multipliers the same notation of the functions $v(\cdot)$, $\theta(\cdot)$ (e.g., $\lambda^{SD}(\cdot) = \left(\lambda_1^{SD}(\cdot), \ldots, \lambda_m^{SD}(\cdot)\right)^\top$ is the Lagrange multipliers vector associated with the steepest common descent direction $v^{SD}(\cdot)$).

Due to the presence of the inverse of the convex combination of the matrices $M_1(\cdot), \ldots, M_m(\cdot)$, i.e., $\left[\sum_{j=1}^{m} \lambda_j M_j(\bar{x})\right]^{-1}$, Problem (2.5) could be much more difficult to handle. On the one hand, if the matrices are different among each other, that is, they depend on the associated objective function, Problem (2.5) is harder to solve w.r.t. (2.3). On the other hand, if $M_j(\bar{x}) = \tilde{M} \succ 0$ for all $j \in \{1, \ldots, m\}$, given (2.6) the difficult term $\left[\sum_{j=1}^{m} \lambda_j M_j(\bar{x})\right]^{-1}$ is substituted by $\tilde{M}^{-1}$. As a consequence, Problem (2.5) becomes a linearly constrained, convex quadratic program which is easy to solve and even more appealing to use w.r.t. (2.3), especially in situations where $n \gg m$.

2.2.1 The Quasi-Newton Approximation Matrices

In many works for MOO [89,91], the BFGS update formula is independently used for all B_j, with $j \in \{1, \ldots, m\}$: given $x_k, x_{k+1} \in \mathbb{R}^n$ being the current and next iterates of an algorithm that employs the Quasi-Newton direction, we have that the matrix B_j is updated as

$$B_j^{k+1} = B_j^k - \frac{B_j^k s_k s_k^\top B_j^k}{s_k^\top B_j^k s_k} + \frac{y_j^k (y_j^k)^\top}{s_k^\top y_j^k},$$

where $s_k = x_{k+1} - x_k$ and $y_j^k = \nabla f_j(x_{k+1}) - \nabla f_j(x_k)$.

We also introduce the formula for updating the inverse of the approximation matrix B_j, which we denote by H_j:

$$H_j^{k+1} = \left(I_n - \rho_j^k y_j^k s_k^\top\right)^\top H_j^k \left(I_n - \rho_j^k y_j^k s_k^\top\right) + \rho_j^k s_k s_k^\top, \tag{2.8}$$

where $\rho_j^k = 1/(s_k^\top y_j^k)$.

Similar to the scalar case [84], for each $j \in \{1, \ldots, m\}$, if $s_k^\top y_j^k > 0$ and $B_j^k \succ 0$, then B_j^{k+1} is positive definite. The same property holds true if

$\{H_j^k\}$ is considered. When the objective functions are strictly convex, the condition $s_k^\top y_j^k > 0$ is always satisfied for any pair (x_{k+1}, x_k) and for each $j \in \{1, \ldots, m\}$. However, this property is not guaranteed to hold in the general case.

2.2.2 On the Use of a Single Approximation Matrix

The use of a single positive definite approximation matrix B was proposed in [1] for the unconstrained smooth MOO setting. As anticipated at the end of Section 2.2, using a unique matrix $\tilde{M} = B$ for each objective function $f_j(\cdot)$, with $j \in \{1, \ldots, m\}$, makes Problem (2.5) easier to solve:

$$\begin{aligned} \max_{\lambda \in \mathbb{R}^m} \; & -\frac{1}{2} \lambda^\top J_F(\bar{x}) B^{-1} J_F(\bar{x})^\top \lambda \\ \text{s.t.} \; & \sum_{j=1}^{m} \lambda_j = 1, \qquad \lambda \geq \mathbf{0}_m. \end{aligned} \tag{2.9}$$

Moreover, the modified Quasi-Newton descent direction can accordingly be computed as

$$v^{MQN}(\bar{x}) = -B^{-1} J_F(\bar{x})^\top \lambda^{MQN}(\bar{x}).$$

The unique matrix B is obtainable as the approximation of a convex combination of matrices. For this purpose, slightly modified BFGS update formulas are introduced in [1]:

$$B^{k+1} = B^k - \frac{B^k s_k s_k^\top B^k}{s_k^\top B^k s_k} + \frac{u_k u_k^\top}{s_k^\top u_k}; \tag{2.10}$$

$$H^{k+1} = \left(I_n - \rho^k u_k s_k^\top\right)^\top H^k \left(I_n - \rho^k u_k s_k^\top\right) + \rho^k s_k s_k^\top, \tag{2.11}$$

with $\rho^k = 1/(s_k^\top u_k)$ and $u_k = \sum_{j=1}^m \lambda_j^{MQN}(x_k) y_j^k$.

Chapter 3

Review of State-of-the-art Gradient-Based Algorithms

After introducing the main concepts of Pareto's theory and the definition of descent direction, in this chapter we report some of the standard gradient-based algorithmic schemes from the multi-objective literature. As anticipated in Chapter 1, we distinguish between *single-point* methods and *front-oriented* approaches. The first ones aim to return a single, hopefully optimal, solution; the second ones try to approximate the entire Pareto front, returning multiple solutions among which an user can choose a posteriori the one with the desired trade-off. Accordingly, this chapter is divided into two sections, each of which is related to a specific algorithm type.

3.1 Single-point Methods

In Algorithm 3.1, the scheme of a generic framework for single-point gradient-based methods is reported. At each iteration k, a descent direction is found solving an instance of Problem (2.3) (Line 4). Then, in Line 5 a step along the direction $v(x_k)$ is found employing a line search technique and, finally, in Line 6 the new iterate x_{k+1} is generated.

Through the next proposition, we state, under the hypothesis of convexity of the feasible set Ω and of convergence to Pareto stationarity, the well-definiteness property of the framework when an ε-Pareto-stationary solution, with $\varepsilon > 0$, is required.

Algorithm 3.1: Generic Framework for Single-point Gradient-Based Methods

Input: $F : \mathbb{R}^n \to \mathbb{R}^m$, Ω feasible set, $x_0 \in \Omega$.
$k = 0$
while x_k *is not Pareto stationary* **do**
  Compute

$$v(x_k) \in \underset{d \in \mathcal{D}(x_k)}{\arg\min} \max_{j=1,\dots,m} \nabla f_j(x_k)^\top d + \frac{1}{2} d^\top M_j(x_k) d$$

  Perform a line search to find α_k
  $x_{k+1} = x_k + \alpha_k v(x_k)$
  $k = k + 1$
return x_k

Proposition 3.1. *Let us assume that Algorithm 3.1 generates a sequence of points $\{x_k\} \subseteq \Omega$, with Ω being a convex feasible set, such that $\{x_k\}$ admits limit points each of which is Pareto-stationary for Problem* (2.1)*. Then, given $\varepsilon > 0$, Algorithm 3.1 finds in a finite number of steps a point which is ε-Pareto-stationary for* (2.1)*.*

Proof. We assume, by contradiction, that Algorithm 3.1 produces an infinite sequence of points $\{x_k\}$ such that, for all k, x_k is not ε-Pareto-stationary for Problem (2.1). Since the method converges to Pareto-stationary points, there exists a subsequence $K \subseteq \{1, 2, \dots\}$ such that

$$\lim_{\substack{k \to \infty \\ k \in K}} x_k = \bar{x}$$

and $\bar{x}$ is Pareto-stationary for (2.1). By Point 3. of Lemma 2.2 and Table 2.1, we thus have that $\theta^{CS}(\bar{x}) = 0$.

Given that, by Point 2. of Lemma 2.4, $\theta^{CS}(\cdot)$ is continuous in Ω, we get that

$$\lim_{\substack{k \to \infty \\ k \in K}} \theta^{CS}(x_k) = 0 > -\varepsilon.$$

Reminding that $\theta^{CS}(x) \leq 0$ for all $x \in \Omega$ (Point 2. of Lemma 2.2), this last result implies that, for sufficiently large values for $k \in K$, the following equation has to hold: $\theta^{CS}(x_k) = \min_{d \in \mathcal{D}(x_k)} \max_{j \in \{1,\dots,m\}} \nabla f_j(x_k)^\top d + \frac{1}{2}\|d\|^2 \geq$

$-\varepsilon$, i.e. by Definition 2.4 x_k is ε-Pareto-stationary for (2.1). Therefore, we get the contradiction and the assertion is proved. □

Depending on the employed descent direction (see Table 2.1) and line search technique, the framework reduces to some classical gradient-based methodologies for multi-objective optimization. Below, we report a brief description of each of them.

3.1.1 Multi-Objective Steepest Descent

Based on the concept of steepest common descent direction (see Table 2.1), the standard *Multi-Objective Steepest Descent* (`MOSD`) algorithm was proposed in [36]. The approach employs a back-tracking *Armijo-type Line Search*, whose description along with its scheme and finite-termination property can be found in Section 3.2.1.

Before proceeding with the `MOSD` convergence property, we need to introduce a reasonable assumption, which will be also used in other contexts of this thesis.

Assumption 3.1. The objective function $F(\cdot)$ has bounded level sets in the multi-objective sense, i.e., the set $\mathcal{L}_F(z) = \{x \in \Omega \mid F(x) \leq z\}$ is bounded for any $z \in \mathbb{R}^m$.

Lemma 3.1 ([36, Theorem 1])**.** *Every accumulation point of the sequence $\{x_k\} \subseteq \mathbb{R}^n$ produced by the `MOSD` algorithm is a Pareto-stationary point. If Assumption 3.1 holds with $\Omega = \mathbb{R}^n$ and $z = F(x_0)$, then the sequence $\{x_k\}$ stays bounded and has at least one accumulation point.*

3.1.2 Multi-Objective Projected Gradient

The *Multi-Objective Projected Gradient* (`MOPG`) method was firstly introduced in [29] and then developed and analyzed in [39, 40]. In addition, the `MOPG` main results were summarized in [38]. The method is an extension of the `MOSD` procedure dealing with constrained problems. In particular, it deals with optimization problems characterized by $\Omega = C \subset \mathbb{R}^n$, with C being a feasible closed and convex set. The considered descent direction in `MOPG` is the constrained steepest common descent one (see Table 2.1), while the employed Armijo-type line search is the same one of `MOSD` (further information about this technique can be found in Section 3.2.1).

We report here two theoretical results of the `MOPG` method.

Lemma 3.2 ([38, Lemma 4.3]). *Let $\{x_k\} \subset \mathbb{R}^n$ be a sequence generated by* `MOPG`*. Then, we have $\{x_k\} \subset \Omega$.*

Lemma 3.3 ([38, Theorem 4.4]). *Every accumulation point, if any, of a sequence $\{x_k\}$ generated by* `MOPG` *is a feasible Pareto-stationary point.*

3.1.3 Newton-type Methods

The considered Newton-type methods deal with unconstrained optimization problems ($\Omega = \mathbb{R}^n$). In particular, we will consider:

- the *Multi-Objective Newton* (`NWT`) method [34];
- the *Multi-Objective Quasi-Newton* (`Q-NWT`) algorithm [89];
- the *Multi-Objective Modified Quasi-Newton* (`MQ-NWT`) approach [1].

We refer the reader to Table 2.1 in order to have more details about the direction type employed in each algorithm. Note that, unlike the last two methodologies, `NWT` only deals with problems characterized by $F(\cdot)$ twice continuously differentiable and strictly convex, since only in this scenario the second derivatives $\nabla^2 f_1(\cdot), \ldots, \nabla^2 f_m(\cdot)$ exist and are positive definite. Regarding the line search techniques employable in these methods, we refer the reader to Section 3.2. Finally, for the sake of brevity, we do not report here the detailed convergence properties of the approaches: these latter ones can be deeply analyzed in the referenced papers.

3.2 Single-point Line Search Techniques

In this section, we provide a brief description of the Armijo-type and Wolfe line searches employable in the single-point methods introduced in Section 3.1.

3.2.1 Armijo-type Line Search

Given a descent direction at the current solution x_k, the idea of the *Armijo-Type Line Search* (`ALS`) [36] is to reduce the step size until we get a sufficient decrease for all the objective functions. The scheme of `ALS` is reported in Algorithm 3.2.

In the next proposition, we report the finite termination property of the line search.

Algorithm 3.2: Armijo-type Line Search (`ALS`)

Input: $F : \mathbb{R}^n \to \mathbb{R}^m$, Ω feasible set, $x_k \in \Omega$, $v(x_k)$ descent direction, $\alpha_0 > 0$, $\delta \in (0,1)$, $\gamma \in (0,1)$.
$\alpha = \alpha_0$
while $x_k + \alpha v(x_k) \notin \Omega \vee F(x_k + \alpha v(x_k)) \not\preceq F(x_k) + \gamma \alpha J_F(x_k) v(x_k)$ **do**
  $\alpha = \delta \alpha$
return α

Proposition 3.2. *If $F(\cdot)$ is continuously differentiable and $J_F(x)v(x) < \mathbf{0}_m$, then there exists some $\varepsilon > 0$, which may depend on x, $v(x)$, γ and Ω, such that*

$$x + tv(x) \in \Omega$$

and

$$F(x + tv(x)) < F(x) + \gamma t J_F(x) v(x)$$

for all $t \in (0, \varepsilon]$.

Proof. Given the hypothesis, by Lemma 4 in [36], we have that there exists some $\bar{\varepsilon} > 0$, which may depend on x, $v(x)$ and γ, such that, for all $t \in (0, \bar{\varepsilon}]$,

$$F(x + tv(x)) < F(x) + \gamma t J_F(x) v(x).$$

Moreover, by Equation (2.3) $v(x) \in \mathcal{D}(x)$ and, then, by (2.2) there exists $\bar{t} > 0$ such that for all $t \in [0, \bar{t}\,]$

$$x + tv(x) \in \Omega.$$

Thus, we get the thesis choosing $\varepsilon = \min\{\bar{\varepsilon}, \bar{t}\,\}$. □

Remark 3.1. By the definition of $D(\cdot,\cdot)$ (2.4), we have that $J_F(x)v(x) \leq \mathbf{1}_m D(x, v(x))$. Moreover, if $M_j(x) \succ 0 \ \forall j \in \{1, \ldots, m\} \ \forall x \in \Omega$, then it follows that $D(x, v(x)) < \theta(x)$. Using Proposition 3.2 and these results, we trivially obtain that, for all $t \in (0, \varepsilon]$,

$$\begin{aligned} F(x + tv(x)) &< F(x) + \gamma t J_F(x) v(x) \\ &\leq F(x) + \mathbf{1}_m \gamma t D(x, v(x)) \\ &< F(x) + \mathbf{1}_m \gamma t \theta(x). \end{aligned}$$

3.2.2 Wolfe Line Search

The *Wolfe conditions* have been extended to MOO in [71]: given the current solution x_k and a descent direction $v(x_k)$, we want to find a step size α satisfying

$$F\left(x_k + \alpha v(x_k)\right) \leq F(x_k) + \mathbf{1}_m \gamma \alpha D\left(x_k, v(x_k)\right),$$

and

$$D\left(x_k + \alpha v(x_k), v(x_k)\right) \geq \sigma D(x_k, v(x_k)),$$

where $\gamma, \sigma \in (0,1)$. In the unconstrained scenario, assuming that $v(x_k)$ is a descent direction for $F(\cdot)$ at x_k and there exists $\mathcal{A} \in \mathbb{R}^m$ such that $F\left(x_k + \alpha v(x_k)\right) \geq \mathcal{A}$ for all $\alpha > 0$, an interval of values exists satisfying these conditions [71, Proposition 3.2]. The theoretical result can be further improved assuming the boundedness of at least one objective function [72, Proposition 1].

To the best of our knowledge, the first Wolfe line search for unconstrained MOO was proposed in [72]. For the sake of brevity, we do not report here nor the scheme or the finite termination property of this methodology; for further information on the topic, we refer the reader to the referenced paper.

Remark 3.2. As mentioned in Section 2.2.1, similar to the scalar case, in Quasi-Newton methods it is crucial to have, for each $j \in \{1, \ldots, m\}$ and iteration k, that $s_k^\top y_j^k > 0$ in order to maintain the positive definiteness of the approximation matrix B_j. Indeed, unless $F(\cdot)$ is component-wise strictly convex, the latter inequality is not guaranteed. In order to overcome this issue, in Quasi-Newton methods for scalar optimization, Wolfe conditions are imposed at each iteration [84]. However, in MOO, even if the Wolfe conditions are satisfied, it may occur that, at the iteration k, $s_k^\top y_j^k \leq 0$ for some $j \in \{1, \ldots, m\}$. In other words, considering that

$$s_k = x_{k+1} - x_k = \alpha_k v^{QN}(x_k), \tag{3.1}$$

we may have that, for some $j \in \{1. \ldots, m\}$,

$$\left[\nabla f_j(x_{k+1}) - \nabla f_j(x_k)\right]^\top v^{QN}(x_k) \leq 0,$$

which can be also re-written in the form

$$\nabla f_j(x_{k+1})^\top v^{QN}(x_k) \leq \nabla f_j(x_k)^\top v^{QN}(x_k).$$

For this reason, a different formula for updating B_j is introduced in [90]. The corresponding update formula for H_j remains similar to (2.8), except that ρ_j^k is now defined as

$$\rho_j^k = \begin{cases} 1/\left(s_k^\top y_j^k\right) & \text{if } s_k^\top y_j^k > 0, \\ 1/\left[D\left(x_{k+1}, s_k\right) - \nabla f_j\left(x_k\right)^\top s_k\right] & \text{otherwise.} \end{cases} \tag{3.2}$$

Using the above update rule, ρ_j^k is proved to be strictly positive even when $s_k^\top y_j^k \leq 0$. Thus, H_j^{k+1} and, consequently, B_j^{k+1} always remain positive definite.

3.3 The Front Steepest Descent Algorithm

Through the years, the `MOSD` procedure (Section 3.1.1) was extended to handle a sequence of sets $\{X_k\}$ of non-dominated points, rather than a sequence of points, aiming to approximate the Pareto front of the optimization problems. In a context like the MOO one, providing multiple and different solutions could be more useful than returning a single one: the user would be indeed free to choose, a posteriori, the solution providing the most appropriate trade-off among many for themselves. An algorithm representing an extension of the `MOSD` procedure in this direction is the *Front Steepest Descent* (`FSD`), firstly introduced in [23]. As mentioned in Chapter 1, `FSD` is not the only *front-oriented* gradient-based algorithm in the literature; however, it was inspirational for most of the works presented in this thesis and, for this reason, we decided to recall its scheme and characteristics in a dedicated, albeit short, section.

Before proceeding with the description of `FSD`, we need to introduce two preliminary concepts, which will be also crucial in the mechanisms of our proposals.

3.3.1 The Partial Descent Direction

As reported in Section 3.3.3, not all the objective functions are necessarily considered in every iteration of `FSD` to define the search direction: considering only a subset of them may be indeed useful to spread the search in the objectives space and to reach the extreme regions of the Pareto front. In this case, the resulting direction is called *steepest partial descent direction*.

Similar to Problem (2.3), we propose a more general formulation to find *partial descent directions*:

$$\min_{d \in \mathcal{D}(\bar{x})} \max_{j \in \mathcal{I}} \nabla f_j(\bar{x})^\top d + \frac{1}{2} d^\top M_j(\bar{x}) d, \tag{3.3}$$

where $\mathcal{I} \subseteq \{1, \ldots, m\}$ is a subset of indices of objectives. Again, if the matrices $M_j(\cdot)$, with $j \in \mathcal{I}$, are positive definite, Problem (3.3) has a unique minimizer $v_{\mathcal{I}}(\bar{x})$. Accordingly, the optimal value of the problem at $\bar{x}$ will be denoted as $\theta_{\mathcal{I}}(\bar{x})$. It is trivial to see that, for all $x \in \Omega$:

- $\theta_{\mathcal{I}}(x) \leq \theta(x) \leq 0$;
- $\theta_{\mathcal{I}}(x) = \theta(x)$ when $\mathcal{I} = \{1, \ldots, m\}$;
- $\theta_{\mathcal{I}}(\cdot)$ inherits the continuity property, if it exists, of $\theta(\cdot)$.

Similar to the steepest common descent direction (Table 2.1), the steepest partial one represents a special case of partial descent direction where we have $\Omega = \mathbb{R}^n$ and $M_j(\cdot) = I_n$ for all $j \in \mathcal{I}$. Moreover, a similar statement can be also applied for the other well-known directions from the MOO literature listed in the referenced table. Throughout the thesis, the partial descent directions will be always distinguished from the common ones by the subscript $\mathcal{I}$ in the symbols $v(\cdot), \theta(\cdot)$ (e.g., the steepest partial descent direction will be denoted as $v_{\mathcal{I}}^{SD}(\cdot)$).

3.3.2 The Front Armijo-Type Line Search

In [23], a *front-oriented* variant of `ALS` (Algorithm 3.2) is introduced; we call it *Front Armijo-Type Line Search* (`FALS`). In order to describe this technique, we need to introduce two additional notations: given a subset $\mathcal{I} \subseteq \{1, \ldots, m\}$ and a set of points $\bar{X} \subset \mathbb{R}^n$, we define

- $F_{\mathcal{I}}(\bar{x})$ as the $|\mathcal{I}|$-dimensional vector with components $f_j(\bar{x})$, with $j \in \mathcal{I}$, and
- $\bar{X}_{\mathcal{I}} \subseteq \bar{X}$ as the set of points that are mutually non-dominated w.r.t. $F_{\mathcal{I}}(\cdot)$, i.e.,

$$\bar{X}_{\mathcal{I}} = \{x \in \bar{X} \mid \nexists y \in \bar{X} \text{ s.t. } F_{\mathcal{I}}(y) \lneqq F_{\mathcal{I}}(x)\}. \tag{3.4}$$

In Algorithm 3.3, we report the scheme of FALS. In this line search procedure, the step size is reduced until a sufficient decrease is reached w.r.t. all the points in $\bar{X}_{\mathcal{I}}$ for at least one of the objective functions $f_j(\cdot)$, with $j \in \mathcal{I}$. FALS can be considered as a weak extension of ALS to the multi-objective case. As it is not required to obtain a sufficient decrease for all the objective functions, employing FALS leads to two consequences: less required computational time and larger values for the step size. These features may be very useful to obtain good and spread Pareto front approximations in a short time.

Algorithm 3.3: Front Armijo-type Line Search (FALS)

Input: $F : \mathbb{R}^n \to \mathbb{R}^m$, $\mathcal{I} \subseteq \{1, \dots, m\}$, $X_{\mathcal{I}}^k \subset \mathbb{R}^n$ set of mutually non-dominated points w.r.t. $F_{\mathcal{I}}(\cdot)$, $x_c \in X_{\mathcal{I}}^k$, $\alpha_0 > 0$, $\delta \in (0,1)$, $\gamma \in (0,1)$.
$\alpha = \alpha_0$
while $\exists y \in X_{\mathcal{I}}^k$ *s.t.* $F_{\mathcal{I}}(y) + \mathbf{1}_{|\mathcal{I}|}\gamma\alpha\theta_{\mathcal{I}}^{SD}(x_c) < F_{\mathcal{I}}(x_c + \alpha v_{\mathcal{I}}^{SD}(x_c))$ **do**
  $\alpha = \delta\alpha$
return α

FALS has a finite termination property, which we recall in the following lemma.

Lemma 3.4. *[23, Proposition 4] Let $\mathcal{I} \subseteq \{1, \dots, m\}$, $x_c \in X_{\mathcal{I}}^k$ be such that $\theta_{\mathcal{I}}^{SD}(x_c) < 0$, i.e., $v_{\mathcal{I}}^{SD}(x_c)$ exists such that*

$$\nabla f_j(x_c)^\top v_{\mathcal{I}}^{SD}(x_c) + \frac{1}{2}\|v_{\mathcal{I}}^{SD}(x_c)\|^2 < 0, \qquad \forall j \in \mathcal{I}.$$

Then $\exists \bar{\alpha} > 0$ such that

$$F_{\mathcal{I}}(y) + \mathbf{1}_{|\mathcal{I}|}\gamma\bar{\alpha}\theta_{\mathcal{I}}^{SD}(x_c) \not< F_{\mathcal{I}}(x_c + \bar{\alpha}v_{\mathcal{I}}^{SD}(x_c)), \quad \forall y \in X_{\mathcal{I}}^k,$$

i.e., the while loop of FALS terminates in a finite number $\bar{h}$ of iterations, returning a value $\bar{\alpha} = \delta^{\bar{h}}\alpha_0$. Furthermore, the produced point $x_c + \bar{\alpha}v_{\mathcal{I}}^{SD}(x_c)$ is not dominated with respect to the set X^k.

3.3.3 Algorithmic Scheme and Properties

After introducing the notion of partial descent direction and the FALS procedure, we are ready to report the scheme of FSD in Algorithm 3.4. In brief, at

each iteration k all the points in the set X^k (Step 5) and any possible subset of objectives $\mathcal{I} \subseteq \{1, \ldots, m\}$ (Step 6) are considered: if x_c is not dominated with respect to $F_{\mathcal{I}}(\cdot)$ and $\theta^{SD}_{\mathcal{I}}(x_c) < 0$, an execution of `FALS` (Algorithm 3.3) is carried out to find a step size along the steepest partial descent direction (Step 8); the new resulting point is then added to the current set of solutions $\hat{X}^k$, while all points that are dominated by it are removed (Step 9).

Algorithm 3.4: Front Steepest Descent (`FSD`)

1 Input: $F : \mathbb{R}^n \to \mathbb{R}^m$, $X^0 \subset \mathbb{R}^n$ set of mutually non-dominated points w.r.t. $F(\cdot)$.
2 $k = 0$
3 **while** *a stopping criterion is not satisfied* **do**
4 $\hat{X}^k = X^k$
5 **forall** $x_c \in X^k$ **do**
6 **forall** $\mathcal{I} \subseteq \{1, \ldots, m\}$ *such that*
- $x_c \in \hat{X}^k_{\mathcal{I}}$ *and*
- $\theta^{SD}_{\mathcal{I}}(x_c) < 0$

7 **do**
8 $\alpha = \texttt{FALS}(F(\cdot), \mathcal{I}, \hat{X}^k_{\mathcal{I}}, x_c)$
9 $\hat{X}^k = \left(\hat{X}^k \cup \left\{x_c + \alpha v^{SD}_{\mathcal{I}}(x_c)\right\}\right) \setminus \left\{y \in \hat{X}^k \mid F\left(x_c + \alpha v^{SD}_{\mathcal{I}}(x_c)\right) \lneqq F(y)\right\}$
10 $X^{k+1} = \hat{X}^k$
11 $k = k + 1$
12 **return** X^k

Now, we shall recall the convergence property of `FSD`. This one is based on the concept of *linked sequence* [67], whose definition is reported below.

Definition 3.1. Let $\left\{X^k\right\}$ be a generated sequence of sets of non-dominated points. We define a linked sequence as a sequence $\{x_{j_k}\}$ such that, for any $k = 1, 2, \ldots$, the point $x_{j_k} \in X^k$ is generated at iteration $k - 1$ by the point $x_{j_{k-1}} \in X^{k-1}$.

Before proceeding with the `FSD` convergence property, we also need to state the following assumption. We report it in a more general way, i.e., consid-

ering a generic feasible set Ω, since it will be also used in other chapters of the thesis.

Assumption 3.2. Given a set of non-dominated points $X^0 \subset \Omega$, there exists $x_0 \in X^0$ such that:

1. x_0 is not Pareto-stationary for Problem (2.1);
2. the set $\overline{\mathcal{L}}_F(x_0) = \bigcup_{j=1}^{m} \{x \in \Omega \mid f_j(x) \leq f_j(x_0)\}$ is compact.

Note that this assumption is stronger than the one required to prove the convergence of the `MOSD` method (Assumption 3.1 with $\Omega = \mathbb{R}^n$). However, as observed in [23], this is reasonable since the stopping criterion of `FALS` (Algorithm 3.3) is weaker than the second one used in `ALS` (Algorithm 3.2).

Lemma 3.5 ([23, Proposition 5])**.** *Let Assumption 3.2 hold with $\Omega = \mathbb{R}^n$. Let $\{X^k\}$ be the sequence of sets of nondominated points produced by Algorithm 3.4. Let $\{x_{j_k}\}$ be a linked sequence, then it admits limit points and every limit point is Pareto-stationary for Problem* (2.1)*.*

Remark 3.3. An improved version of Algorithm 3.3 was also proposed in [23], which is based on an extrapolation strategy and allows to possibly obtain many non-dominated solutions along the search direction. When used within Algorithm 3.4, the extrapolation technique does not alter theoretical convergence results, but the resulting algorithm is reported to be significantly more effective.

Finally, the `FSD` algorithm can be adapted to handle box-constrained optimization problems. We call the adaptation *Front Projected Gradient* (`FPG`) and we provide a full description of it in Appendix A, along with feasibility and convergence properties.

Chapter 4

A Memetic Procedure for Global Multi-Objective Optimization

In this chapter [1], we consider multi-objective optimization problems over a box, i.e.,

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \; & F(x) = (f_1(x), \ldots, f_m(x))^\top \\ \text{s.t. } & x \in [l, u], \end{aligned}$$

where $l, u \in \mathbb{R}^n$ are such that, for all $i \in \{1, \ldots, n\}$, $l_i \leq u_i$. In line with Problem (2.1), we denote the feasible set as $\Omega = \{x \in \mathbb{R}^n \mid x \in [l, u]\}$. Several computational approaches to solve these problems have been proposed in the literature, that broadly fall into two main classes: evolutionary algorithms (EAs), which are usually very good at exploring the feasible region and retrieving good solutions even in the nonconvex case, and descent methods, which excel in efficiently approximating good quality solutions. However, each of these two classes also has non-negligible practical drawbacks: EAs have no theoretical convergence property [34] and are usually expensive [49, 66, 94]; on the other hand, descent algorithms often produce suboptimal solutions when starting from non carefully chosen points and are thus not suitable for highly non-convex problems. Here, we propose a

[1] Part of the content of this chapter has been published as "A memetic procedure for global multi-objective optimization" in *Mathematical Programming Computation, 2023* [61].

new memetic method which combines the good features of both. Memetic algorithms are particularly successful strategies in scalar optimization, combining population-based techniques (either heuristic and/or genetic ones) and local search steps [16, 46, 47, 68, 69, 79]. In the case of MOO, this idea has only superficially been considered. Indeed, we can find approaches that are mostly application-specific [103] or that employ heuristic [55, 62, 64, 73], meta-heuristic [2, 101], stochastic [28, 33] or scalarization-based [49, 94, 100] local search steps. Even the few proposed strategies employing gradient information for the local search steps do not exploit the concept of common descent directions. Rather, convex combinations of gradients are generated and exploited in various ways [10, 14, 62, 66, 93]. Our proposal, which we call *Non-dominated Sorting Memetic Algorithm* (`NSMA`), combines an evolutionary approach, i.e., `NSGA-II`, which represents the de facto standard at least popularity-wise for unconstrained and bound-constrained MOO, with MO descent methods. In particular, in order to be executed in combination with `NSGA-II`, a new front-oriented descent method is presented and theoretically analyzed.

4.1 Preliminaries: `NSGA-II`

The *Nondominated Sorting Genetic Algorithm II* (`NSGA-II`) is a multi-objective evolutionary algorithm that was proposed in [26]. In particular, `NSGA-II` is a genetic algorithm that creates a mating pool by combining the parent and offspring populations and selecting the best N solutions. In this section, we review the main characteristics of `NSGA-II`. For a deeper understanding of the algorithm mechanisms, the reader is referred to the original work [26].

We report the main steps of `NSGA-II` in Algorithm 4.1.

`NSGA-II` deals with a fixed size population (N solutions) and takes as input an initial population X^0. For the sake of clarity, from now on we consider X^0 as a set composed by N feasible solutions. However, we want to remark two facts.

- Starting with a population X^0 composed by N points is not necessary: if the population is smaller/bigger, after the first iteration it is increased/reduced in order to get exactly N solutions in it.
- `NSGA-II` can also manage unfeasible points. However, since in our work we address bound constrained problems, and the genetic operators en-

Algorithm 4.1: Nondominated Sorting Genetic Algorithm II (NSGA-II)

1 Input: $F : \mathbb{R}^n \to \mathbb{R}^m$, Ω feasible set, $X^0 \subset \Omega$, N population size.
2 $k = 0$
3 $\hat{X}^0 = X^0$
4 $\hat{R}^0, \hat{C}^0 = \texttt{getMetrics}(\hat{X}^0)$
5 $X^0, R^0, C^0 = \texttt{getSurvivors}(\hat{X}^0, \hat{R}^0, \hat{C}^0, N)$
6 **while** *a stopping criterion is not satisfied* **do**
7 $\quad P^k = \texttt{getParents}(X^k, R^k, C^k)$
8 $\quad O^k = \texttt{crossover}(P^k, l, u)$
9 $\quad \tilde{O}^k = \texttt{mutation}(O^k, l, u)$
10 $\quad \hat{X}^{k+1} = X^k \cup \tilde{O}^k$
11 $\quad \hat{R}^{k+1}, \hat{C}^{k+1} = \texttt{getMetrics}(\hat{X}^{k+1})$
12 $\quad X^{k+1}, R^{k+1}, C^{k+1} = \texttt{getSurvivors}(\hat{X}^{k+1}, \hat{R}^{k+1}, \hat{C}^{k+1}, N)$
13 $\quad k = k + 1$
14 **return** X^k

sure that after the first iteration no point in the population violates the bound constraints, we assume that X^0 is only composed by feasible points.

The core idea of the algorithm is that during an iteration:

- the parents are chosen among the current solutions (Line 7);
- N offsprings are created from the parents through the `crossover` operator (Line 8);
- the offsprings are mutated using the `mutation` function (Line 9);
- a new population of $2N$ solutions is created merging the current population with the offsprings (Line 10);
- by the function `getMetrics` (Section 4.1.1) scores are associated to the members (Line 11);
- by the function `getSurvivors` (Section 4.1.3) only the best N points (survivors) are selected and maintained (Line 12).

The `crossover` and `mutation` operators have a crucial role in the `NSGA-II` mechanisms. The `crossover` operator aim is the creation of offsprings that inherit (hopefully the best) features of the parents. The `mutation` operator introduces some random changes in the offsprings. This latter one could be useful when we want to spread our search in the objectives space as much as possible. For a more detailed and technical explanation about these two operators, we again refer the reader to [26]. We want to remark here that the `NSGA-II` mechanisms ensure that there are no duplicates among the offsprings and any offspring is not a duplicate of any point in the current population. At the end of the algorithm execution, the current population X^k is returned.

In the next subsections, we provide other details of the algorithm that are useful for our purposes.

4.1.1 Metrics

In this section, we explain the metrics used in the `NSGA-II` mechanisms (computed in the `getMetrics` function in Lines 4-11 of Algorithm 4.1). In particular, these scores are used to select the parents and the survivors.

The first one is the *ranking* (R), which leads to a splitting of the population in different domination levels. Briefly, if a point has rank 0, it means that it is not dominated by any point in X^k w.r.t. $F(\cdot)$. If it has rank 1, it is dominated by some of the points with rank 0, but it is not w.r.t. any other point with rank equal to or greater than 1. In order to obtain the ranking values, a fast sorting approach is employed, which is one of the strength elements of the `NSGA-II` algorithm.

The second considered metric is *crowding distance* (C). It is useful to get an estimate of the density of solutions surrounding a particular point in the population. Having a high crowding distance indicates that the point is in a poorly populated area of the objectives space, and maintaining it in the population may likely lead to a spread Pareto front. Note that for each point this metric is calculated with respect to the solutions with the same ranking value.

We again refer the reader to the original paper [26] for the rigorous definition of the metrics.

4.1.2 Parents Selection

In the `getParents` function, pairs of parents are randomly chosen among the solutions in X^k. Then, considering a pair, only one of the two points is selected by *binary tournament*. In this latter one, the solutions are compared in the following way.

- The point with the lowest rank is preferred.
- If the ranking value is the same for both points, the one with the highest crowding distance is chosen.
- In the unlikely case in which the crowding distance values are equal too, a random choice is done.

The selected point will be used with a parent chosen from another pair in the `crossover` function in order to create offsprings.

This approach of comparing the solutions is also used in the `getSurvivors` function.

4.1.3 Selection Operation

After getting the offsprings through the `crossover` and `mutation` operators, the new population is composed by $2N$ solutions. The aim of the `getSurvivors` function (Lines 5-12 of Algorithm 4.1) is to select and maintain the best N solutions. As in the `getParents` function, the selection is based on the ranking and the crowding distance.

- The set composed by the $2N$ points is initially sorted based on the ranking.
- The solutions with the same rank are sorted based on the crowding distance.
- The first N points are chosen as the best ones.

4.2 Non-dominated Sorting Memetic Algorithm

In this section, we introduce our novel memetic algorithm for bound-constrained MOO problems, which we call *Non-dominated Sorting Memetic Algorithm*

(NSMA). We first show and describe the algorithmic scheme. Then, we formally introduce the *Front Multi-Objective Projected Gradient* (FMOPG) algorithm, which is the descent method used within the NSMA and for which we also provide a rigorous theoretical analysis.

4.2.1 Algorithmic Scheme

The scheme of NSMA is reported in Algorithm 4.2. Basically, the structure of the proposed algorithm is similar to that of NSGA-II, from which we also inherit all the genetic operators. The main differences between the two methods are constituted by three new operations:

- `getSurrogateBounds` (Line 8);
- `getCrowdingDistanceThreshold` (Line 13);
- `optimizePopulation` (Line 16).

In the next subsections, we give a detailed description of these three new functions.

4.2.1.1 Estimating Surrogate Bounds

When the addressed problem is characterized by a feasible region that is particularly large w.r.t. the one(s) where the Pareto set lies, the NSGA-II algorithm turns out to be slow at obtaining a good approximation of the Pareto front. This issue occurs because of the `crossover` and, above all, of the `mutation` operator. Random mutations over a large search area lead from the very first iterations to a population which is overly disperse and far from optimality. In such a scenario, even the effectiveness of the `crossover` operator might be compromised: some parents might have extremely bad features. As a consequence, NSGA-II may exhibit a performance slowdown.

In NSMA, this issue is solved using surrogate bounds for the `crossover` and the `mutation` operators instead of the original ones. These bounds are obtained using the `getSurrogateBounds` function, which we report in Algorithm 4.3. The surrogate bounds are computed using the current population and a shift value s_h. This latter parameter is employed to progressively enlarge the region where the population can be distributed: a greater value leads to a bigger enlargement.

Algorithm 4.2: Non-dominated Sorting Memetic Algorithm (`NSMA`)

Input: $F : \mathbb{R}^n \to \mathbb{R}^m$, Ω feasible set, $X^0 \subset \Omega$, N population size, $s_h \in \mathbb{R}^+$, $q \in [0, 1]$, $n_{opt} \in \mathbb{N}^+$, $\{\varepsilon_t\} \subset \mathbb{R}_0^+$ decreasing sequence.
$k = 0$, $t = 0$
$\hat{X}^0 = X^0$
$\hat{R}^0, \hat{C}^0 = \texttt{getMetrics}(\hat{X}^0)$
$X^0, R^0, C^0 = \texttt{getSurvivors}(\hat{X}^0, \hat{R}^0, \hat{C}^0, N)$
while *a stopping criterion is not satisfied* **do**
  $P^k = \texttt{getParents}(X^k, R^k, C^k)$
  $l_k^s, u_k^s = \texttt{getSurrogateBounds}(X^k, l, u, s_h)$
  $O^k = \texttt{crossover}(P^k, l_k^s, u_k^s)$
  $\tilde{O}^k = \texttt{mutation}(O^k, l_k^s, u_k^s)$
  $\hat{X}^{k+1} = X^k \cup \tilde{O}^k$
  $\hat{R}^{k+1}, \hat{C}^{k+1} = \texttt{getMetrics}(\hat{X}^{k+1})$
  $\bar{c}_{k+1} = \texttt{getCrowdingDistanceThreshold}(\hat{X}^{k+1}, \hat{R}^{k+1}, \hat{C}^{k+1}, q)$
  $X^{k+1}, R^{k+1}, C^{k+1} = \texttt{getSurvivors}(\hat{X}^{k+1}, \hat{R}^{k+1}, \hat{C}^{k+1}, N)$
  if k ***mod*** $n_{opt} = 0$ **then**
    $X^{k+1}, R^{k+1}, C^{k+1} =$ $\texttt{optimizePopulation}(F(\cdot), \Omega, X^{k+1}, R^{k+1}, C^{k+1}, \bar{c}_{k+1}, \varepsilon_t, N)$
    $t = t + 1$
  $k = k + 1$
return X^k

Ideally, the exploration starts considering only a small (local) portion of the feasible area, which is defined by the initial population and the shift value s_h. In this way, the points cannot be moved by the `crossover` and the `mutation` operators too far away in the feasible set. At each following iteration, new surrogate bounds are computed to enlarge the search space.

After a number of iterations, it may happen that the surrogate bounds cover a bigger region than the one defined by the original bounds. In such case, the search goes on over the entire feasible set.

Algorithm 4.3: `getSurrogateBounds`

Input: $X^k \subset \Omega$, $l, u \in \mathbb{R}^n$ lower and upper bounds, $s_h \in \mathbb{R}^+$.
for $i = 1, \ldots, n$ **do**
  $(l_k^s)_i = \max\left\{l_i, \min_{x \in X^k}\{x_i\} - s_h\right\}$
  $(u_k^s)_i = \min\left\{u_i, \max_{x \in X^k}\{x_i\} + s_h\right\}$
return l_k^s, u_k^s

4.2.1.2 Identifying Exploration Candidates

Similarly as in memetic approaches for scalar optimization, performing local searches starting from each point in a population usually turns out to be inefficient. In fact, a great computational effort is required to optimize many points that in the end do not lead to good solutions.

In the case of `NSMA`, one may think of only performing local searches for the rank-0 points. However, this idea is inefficient too: during the last iterations, most, if not all, the points are likely to be associated with a ranking value equal to 0. Furthermore, many of these points could be in a high density area of the Pareto front and, therefore, optimizing all of them could be a waste of computational time.

The issue is solved by choosing to optimize the rank-0 points associated with an high crowding distance. As already remarked in Section 4.1.1, such points are in a poorly populated area of the objectives space. Therefore, optimizing them, we still contribute to obtain a better approximation of the Pareto front, since they are rank-0 points, and, at the same time, we have the possibility to populate a low density area, leading to a better spread Pareto front.

Through the `getCrowdingDistanceThreshold` function, which we report in Algorithm 4.4, we retrieve the q–quantile of the crowding distances of the rank-0 points in $\hat{X}^{k+1}$. We denote by $\bar{c}_{k+1}$ this quantity: only the rank-0 points associated with a crowding distance greater than or equal to $\bar{c}_{k+1}$ will be optimized through the `FMOPG` algorithm. Smaller values for the parameter q lead to the optimization of a greater number of points.

As stated in [26], some points will be associated to a crowding distance equal to $+\infty$. These points are considered the extreme solutions of the Pareto front w.r.t. a specific objective function. For this reason, they are

Algorithm 4.4: `getCrowdingDistanceThreshold`

1 Input: $\hat{X}^{k+1} \subset \Omega$, $\hat{R}^{k+1}, \hat{C}^{k+1} \in \mathbb{R}^{|\hat{X}^{k+1}|}$ metrics vectors, $q \in [0, 1]$.
2 $\bar{C}^{k+1} = \{\hat{c}_p \in \hat{C}^{k+1} | \hat{r}_p = 0 \wedge \hat{c}_p < +\infty\}$
3 **if** $\bar{C}^{k+1} \neq \emptyset$ **then**
4 Let $\bar{c}_{k+1}$ be the q–quantile of the set $\bar{C}^{k+1}$
5 **else**
6 $\bar{c}_{k+1} = +\infty$
7 **return** $\bar{c}_{k+1}$

always used as starting solutions for local searches, since they could lead to a wider Pareto front approximation.

4.2.1.3 Local Searches by Multi-Objective Descent

In the `optimizePopulation` function, which we report in Algorithm 4.5, the `FMOPG` method is employed to refine the population by performing local searches. This function is the core of our memetic approach: it allows to combine the typical features of descent methods with the genetic operators of `NSGA-II`.

In order to be optimized through `FMOPG` w.r.t. a subset of indices of objectives $\mathcal{I} \subseteq \{1, \ldots, m\}$, a point x_p must satisfy the following conditions.

- Its rank must be 0 and its crowding distance must be greater than or equal to $\bar{c}_{k+1}$ (Line 5a.). These requirements are already discussed in Section 4.2.1.2.
- It must belong to $\hat{X}_{\mathcal{I}}^{k+1}$, which is the set of mutually non-dominated points w.r.t. $F_{\mathcal{I}}(\cdot)$ contained in $\hat{X}^{k+1}$ (Line 5b.). A formal definition of this set can be found in Equation (3.4). Trying to optimize points which are not contained in $\hat{X}_{\mathcal{I}}^{k+1}$ could be useless, since we have no guarantee to reach a non-dominated point w.r.t. $F_{\mathcal{I}}(\cdot)$.
- It must not be Pareto-stationary w.r.t. $F_{\mathcal{I}}(\cdot)$ (Line 5c.).

If the point satisfies all these requirements, it is used as starting solution in the `FMOPG` algorithm. Along with it, the set $\hat{X}_{\mathcal{I}}^{k+1}$ is used as input for the algorithm. `FMOPG` returns the set of produced solutions, which are collected in the set $\tilde{X}^p$ and inserted in the set $\hat{X}^{k+1}$.

Algorithm 4.5: `optimizePopulation`

Input: $F : \mathbb{R}^n \to \mathbb{R}^m$, Ω feasible closed and convex set, $X^{k+1} \subset \Omega$, $R^{k+1}, C^{k+1} \in \mathbb{R}^{|X^{k+1}|}$ metrics vectors, $\bar{c}_{k+1}$ crowding distance threshold, $\varepsilon_t \in \mathbb{R}_0^+$, N population size.
$\hat{X}^{k+1} = X^{k+1}$
for $p = 1, \ldots, |X^{k+1}|$ **do**
  for $\mathcal{I} \in 2^{\{1,\ldots,m\}}$ **do**
    if x_p *is such that*
      a. $r_p = 0$, $c_p \geq \bar{c}_{k+1}$
      b. $x_p \in \hat{X}_{\mathcal{I}}^{k+1}$
      c. $\theta_{\mathcal{I}}^{CS}(x_p) < 0$ *(see Table 2.1)*
    then
      $\tilde{X}^p = \texttt{FMOPG}(F(\cdot), \Omega, \mathcal{I}, \hat{X}_{\mathcal{I}}^{k+1}, x_p, \varepsilon_t)$
      $\hat{X}^{k+1} = \hat{X}^{k+1} \cup \tilde{X}^p$
$\hat{R}^{k+1}, \hat{C}^{k+1} = \texttt{getMetrics}(\hat{X}^{k+1})$
$X^{k+1}, R^{k+1}, C^{k+1} = \texttt{getSurvivors}(\hat{X}^{k+1}, \hat{R}^{k+1}, \hat{C}^{k+1}, N)$
return $X^{k+1}, R^{k+1}, C^{k+1}$

Lastly, the new population $\hat{X}^{k+1}$ is reduced in order to have exactly N survivors. This last operation is performed through the `getMetrics` (Section 4.1.1) and the `getSurvivors` (Section 4.1.3) functions of the `NSGA-II` algorithm.

4.2.2 The Front Multi-Objective Projected Gradient Algorithm

The *Front Multi-Objective Projected Gradient* (`FMOPG`) algorithm is the descent method used in our memetic approach. In particular, it is a variant of the `MOPG` method (Section 3.1.2).

4.2.2.1 Algorithmic Scheme

We report the scheme of `FMOPG` in Algorithm 4.6.

The main difference between `FMOPG` and `MOPG` is the following: while in

Algorithm 4.6: Front Multi-Objective Projected Gradient (`FMOPG`)

1 Input: $F : \mathbb{R}^n \to \mathbb{R}^m$, Ω feasible closed and convex set, $\mathcal{I} \subseteq \{1, \ldots, m\}$, $X^0 \subset \Omega$, $x_0 \in X^0$.
2 $k = 0$
3 **while** x_k *is not Pareto-stationary w.r.t.* $F_{\mathcal{I}}(\cdot)$ **do**
4 Compute

$$d_k^{\mathcal{I}} = \arg\min_{d \in \mathbb{R}^n} \max_{j \in \mathcal{I}} \nabla f_j(x_k)^\top d + \frac{1}{2}\|d\|^2 \quad \text{s.t. } x_k + d \in \Omega \tag{4.1}$$

5 Let $\theta_k^{\mathcal{I}}$ the optimal value of Problem (4.1) at x_k
6 $\alpha_k = \texttt{B-FALS}(F(\cdot), \Omega, \mathcal{I}, X_{\mathcal{I}}^k, x_k, d_k^{\mathcal{I}}, \theta_k^{\mathcal{I}})$
7 $x_{k+1} = x_k + \alpha_k d_k^{\mathcal{I}}$
8 $X^{k+1} = X^k \cup \{x_{k+1}\}$
9 $k = k + 1$
10 **return** sequence $\{x_k\}$

the original algorithm the current point x_k is only optimized w.r.t. itself, in `FMOPG` it is also w.r.t. the set of points in which it is contained. At each iteration, the direction at the solution x_k is found solving an instance of Problem (4.1); by Problem (2.3) and Table 2.1, it is trivial to see that if $\theta_{\mathcal{I}}^{CS}(x_k) < 0$ thus $\theta_k^{\mathcal{I}} < 0$, i.e., $d_k^{\mathcal{I}}$ is a feasible and descent direction at x_k. Then, in Line 6 a step size α_k is calculated by the `B-FALS` procedure, whose description is reported in Appendix A. In brief, `B-FALS` is an extension of the `FALS` technique (Algorithm 3.3) for the bound-constrained MOO setting: the only added requirement w.r.t. the original procedure is that the step size must lead to a point that is also feasible. Given the direction and the step size, a new point x_{k+1} is finally obtained (Line 7). This latter one is inserted in the set X^k, leading to a new set X^{k+1} (Line 8).

The `FMOPG` algorithm iterates until the current solution x_k is Pareto-stationary w.r.t. $F_{\mathcal{I}}(\cdot)$. At the end, the method returns the sequence of points $\{x_k\}$ generated during the iterations. Indeed, considering the stopping conditions of `B-FALS`, we have no guarantee that, for all k, the point x_{k+1} dominates x_k w.r.t. $F_{\mathcal{I}}(\cdot)$. So, every point produced by `FMOPG` could be useful to obtain good and spread Pareto front approximations.

Finally, note that the FMOPG algorithm is called by the optimizePopulation function with an additional parameter ε_t (Line 7 of Algorithm 4.5). In fact, FMOPG is executed using ε-Pareto-stationarity as stopping condition. In NSMA (Algorithm 4.2), we consider a decreasing sequence $\{\varepsilon_t\} \subset \mathbb{R}_0^+$. So, during the iterations, we get closer and closer to the Pareto-stationarity.

4.2.2.2 Algorithm Analysis

In this section, we provide a rigorous analysis of the FMOPG algorithm from a theoretical perspective. The following analysis is crucial to state the convergence properties of FMOPG. These latter ones are crucial to guarantee that local searches within NSMA stop in finite time and, thus, that the overall algorithm is well-defined.

Before proceeding, we need to state an assumption.

Assumption 4.1. Let $\mathcal{I} \subseteq \{1, \dots, m\}$, $X^0 \subset \Omega$ be a set of feasible points and $x_0 \in X^0$. There does not exist a point $y_0 \in X^0$ that dominates x_0 w.r.t. $F_{\mathcal{I}}(\cdot)$, i.e., $x_0 \in X^0_{\mathcal{I}}$.

This assumption is reasonable since a point x_p to be optimized through FMOPG must be non-dominated w.r.t. $F_{\mathcal{I}}(\cdot)$ (Section 4.2.1.3).

We begin by characterizing the points produced by the FMOPG algorithm.

Proposition 4.1. *Consider a generic iteration k of FMOPG. Let $\mathcal{I} \subseteq \{1, \dots, m\}$, X^k be a set of feasible points and $x_k \in X^k$. Assume that x_k is not dominated by any point in X^k w.r.t. $F_{\mathcal{I}}(\cdot)$. Then, B-FALS returns a step size $\alpha_k > 0$ such that the point $x_{k+1} = x_k + \alpha_k d_k^{\mathcal{I}}$ is feasible and not dominated by any point in X^{k+1} w.r.t. $F_{\mathcal{I}}(\cdot)$.*

Proof. The B-FALS algorithm is performed from $x_k \in X^k_{\mathcal{I}}$, with $\theta_k^{\mathcal{I}} < 0$ (Step 5), along a constrained partial descent direction $d_k^{\mathcal{I}}$. Then, from Proposition A.1, B-FALS terminates in a finite number of steps and returns a step size $\alpha_k > 0$ such that the point $x_{k+1} = x_k + \alpha_k d_k^{\mathcal{I}}$ has the following properties:

- $x_{k+1} \in \Omega$;
- x_{k+1} is not dominated by any other point in X^k w.r.t. $F_{\mathcal{I}}(\cdot)$.

Since $X^{k+1} = X^k \cup \{x_{k+1}\}$, the assertion is finally proved. □

Remark 4.1. Since the point x_{k+1} induced by the step size produced by B-FALS is not dominated by any point in X^{k+1} w.r.t. $F_{\mathcal{I}}(\cdot)$, we can easily conclude that the new point is also not dominated w.r.t. all the objectives.

Given Proposition 4.1, we can state the following corollary.

Corollary 4.1. *Let Assumption 4.1 hold with $\mathcal{I} \subseteq \{1, \ldots, m\}$, the set X^0 and the point x_0. Then, the sequence of sets $\{X^k\}$ and the sequence of points $\{x_k\}$ generated by* FMOPG *are such that for all $k = 0, 1, \ldots$, x_k is feasible and not dominated by any point in X^k w.r.t. $F_{\mathcal{I}}(\cdot)$.*

Proof. The assertion straightforwardly follows if the assumptions of Proposition 4.1 are satisfied at every iteration k of the algorithm.

When $k = 0$, this is guaranteed by Assumption 4.1. The case of a generic iteration k simply follows by induction from Proposition 4.1 itself. □

In order to state the convergence property of FMOPG, from this point forward we also suppose that Assumption 3.2 holds. As already observed in Section 3.3 for the FSD algorithm, this assumption is stronger than the one required to prove convergence of the MOPG method (Assumption 3.1), but still reasonable since the second stopping criterion of B-FALS (Appendix A) is weaker than the second one used in ALS (Algorithm 3.2).

Proposition 4.2. *Let Assumptions 3.2-4.1 hold with $\mathcal{I} \subseteq \{1, \ldots, m\}$, the set X^0 and the point x_0. Let $\{x_k\}$ be the sequence of points generated by* FMOPG. *Then $\{x_k\}$ admits limit points and every limit point is Pareto-stationary considering the objectives $f_j(\cdot)$, with $j \in \mathcal{I}$.*

Proof. Firstly, we prove that the sequence $\{x_k\}$ admits limit points. Since $x_0 \in X^k$ for all k, Corollary 4.1 guarantees that, for each k, $x_k \in \Omega$ and there exists an index $j(x_k) \in \mathcal{I}$ such that $f_{j(x_k)}(x_k) \leq f_{j(x_k)}(x_0)$. So,

$$x_k \in \{x \in \Omega \mid f_{j(x_k)}(x) \leq f_{j(x_k)}(x_0)\}$$

and, therefore, $x_k \in \overline{\mathcal{L}}_F(x_0)$, $\forall k$. Assumption 3.2 assures that the sequence $\{x_k\}$ is bounded. Hence, this latter one admits limit points: we can consider a subsequence $K \subseteq \{1, 2, \ldots\}$ such that

$$\lim_{\substack{k \to \infty \\ k \in K}} x_k = \bar{x}.$$

It is trivial to see that, by Lemma 2.2 and the definition of $\bar{\theta}^{\mathcal{I}}$ (Step 5), $\bar{x}$ is Pareto-stationary w.r.t. $F_{\mathcal{I}}(\cdot)$ if and only if $\bar{\theta}^{\mathcal{I}} = 0$. By contradiction, we assume that $\bar{x}$ is not Pareto-stationary w.r.t. $F_{\mathcal{I}}(\cdot)$: there exists $\bar{\varepsilon} > 0$ such that

$$\theta_k^{\mathcal{I}} \leq -\bar{\varepsilon} < 0, \quad \forall k \in K. \tag{4.2}$$

Next, we want to prove the following statement:

$$\lim_{\substack{k\to\infty \\ k\in K}} \alpha_k \theta_k^{\mathcal{I}} = 0. \tag{4.3}$$

Again, by contradiction, we assume that the assertion is not true: there exists a subsequence $\bar{K} \subseteq K$ and $\bar{\eta} > 0$ such that

$$\alpha_k \theta_k^{\mathcal{I}} \leq -\bar{\eta} < 0, \quad \forall k \in \bar{K}. \tag{4.4}$$

Recalling Proposition A.1 and Corollary 4.1, for all $k \in \bar{K}$, `B-FALS` returns in a finite number of iterations a step size α_k such that

$$x_{k+1} = x_k + \alpha_k d_k^{\mathcal{I}} \in \Omega \tag{4.5}$$

and $F_{\mathcal{I}}(y_k) + \mathbf{1}_{|\mathcal{I}|}\gamma\alpha_k\theta_k^{\mathcal{I}} \not\leq F_{\mathcal{I}}(x_{k+1})$, for all $y_k \in X^k$. By using Equation (4.4), we obtain that, for all $k \in \bar{K}$ and for all $y_k \in X^k$,

$$F_{\mathcal{I}}(y_k) - \mathbf{1}_{|\mathcal{I}|}\gamma\bar{\eta} \not\leq F_{\mathcal{I}}(x_{k+1}). \tag{4.6}$$

Since $\gamma > 0$ and $\bar{\eta} > 0$, we have that $-\gamma\bar{\eta} < 0$.
Since $X^k = X^0 \cup \{x_1\} \cup \ldots \cup \{x_k\}$ and $x_0 \in X^0$, it simply follows that, for all $k \in \bar{K}$, $F_{\mathcal{I}}(x_0) - \mathbf{1}_{|\mathcal{I}|}\gamma\bar{\eta} \not\leq F_{\mathcal{I}}(x_{k+1})$. Therefore, for all $k \in \bar{K}$, there exists $j_k \in \mathcal{I}$ such that $f_{j_k}(x_0) > f_{j_k}(x_0) - \gamma\bar{\eta} \geq f_{j_k}(x_{k+1})$, and, then, considering also Equation (4.5),

$$x_{k+1} \in \overline{\mathcal{L}}_F(x_0). \tag{4.7}$$

Moreover, let us consider $k_1, k_2 \in \bar{K}$, with $k_1 < k_2$. By the instructions of the algorithm, we know that $x_{k_1+1} \in X^{k_2}$. Thus, from Equation (4.6), we know that $F_{\mathcal{I}}(x_{k_1+1}) - \mathbf{1}_{|\mathcal{I}|}\gamma\bar{\eta} \not\leq F_{\mathcal{I}}(x_{k_2+1})$. Therefore, for any pair $k_1, k_2 \in \bar{K}$, with $k_1 < k_2$, there exists $j_{k_2} \in \mathcal{I}$ such that

$$f_{j_{k_2}}(x_{k_1+1}) - \gamma\bar{\eta} \geq f_{j_{k_2}}(x_{k_2+1}). \tag{4.8}$$

Equations (4.7) and (4.8) imply that we have an infinite sequence $\{F_{\mathcal{I}}(x_{k+1})\}_{k\in\bar{K}}$, with $x_{k+1} \in \overline{\mathcal{L}}_F(x_0)$, where any pair of points is at a distance not smaller than $\gamma\bar{\eta}$ from each other. Thus, the set

$$Z = \{z \in \mathbb{R}^m \mid z = F(x_{k+1}), x_{k+1} \in \overline{\mathcal{L}}_F(x_0), k \in \bar{K}\}$$

is not compact. This last statement and the continuity of $F(\cdot)$ contradict Assumption 3.2, since the image of a compact set under a continuous map should be compact. Thus, Equation (4.3) holds.

Recalling Equation (4.2), from Equation (4.3) we obtain the following statement:

$$\lim_{\substack{k\to\infty \\ k\in K}} \alpha_k = 0.$$

Given this limit, we can consider sufficiently large values for $k \in K$ such that

$$\alpha_k < \frac{\alpha_k}{\delta} \leq 1. \tag{4.9}$$

By the convexity of Ω and the definition of the direction $d_k^{\mathcal{I}}$ (4.1), Equation (4.9) implies that the point $x_k + (\alpha_k/\delta)d_k^{\mathcal{I}} \in \Omega$. Therefore, for sufficiently large values for $k \in K$, the stopping conditions of `B-FALS` imply that there exists a point $y_k \in X^k$ such that

$$F_{\mathcal{I}}(y_k) + \mathbf{1}_{|\mathcal{I}|}\gamma\frac{\alpha_k}{\delta}\theta_k^{\mathcal{I}} < F_{\mathcal{I}}\left(x_k + \frac{\alpha_k}{\delta}d_k^{\mathcal{I}}\right). \tag{4.10}$$

Considering Corollary 4.1 and Equation (4.10) respectively, we have that an index $j(x_k) \in \mathcal{I}$ exists such that $f_{j(x_k)}(x_k) + \gamma\frac{\alpha_k}{\delta}\theta_k^{\mathcal{I}} \leq f_{j(x_k)}(y_k) + \gamma\frac{\alpha_k}{\delta}\theta_k^{\mathcal{I}}$ and $f_{j(x_k)}(y_k) + \gamma\frac{\alpha_k}{\delta}\theta_k^{\mathcal{I}} < f_{j(x_k)}\left(x_k + \frac{\alpha_k}{\delta}d_k^{\mathcal{I}}\right)$. Since the set $\mathcal{I}$ is finite, we can consider a subsequence $\bar{K} \subseteq K$ such that, for sufficiently large values for $k \in \bar{K}$, $j(x_k) = \hat{j}$ and, combining the two above inequalities,

$$f_{\hat{j}}\left(x_k + \frac{\alpha_k}{\delta}d_k^{\mathcal{I}}\right) - f_{\hat{j}}(x_k) > \gamma\frac{\alpha_k}{\delta}\theta_k^{\mathcal{I}}.$$

Using the Mean-value Theorem, we have that

$$f_{\hat{j}}\left(x_k + \frac{\alpha_k}{\delta}d_k^{\mathcal{I}}\right) - f_{\hat{j}}(x_k) = \frac{\alpha_k}{\delta}\nabla f_{\hat{j}}(\xi_k)^\top d_k^{\mathcal{I}},$$

with $\xi_k = x_k + t_k\frac{\alpha_k}{\delta}d_k^{\mathcal{I}}$, $t_k \in (0,1)$. Then, we can write $\nabla f_{\hat{j}}(\xi_k)^\top d_k^{\mathcal{I}} > \gamma\theta_k^{\mathcal{I}}$, from which we can state that

$$\nabla f_{\hat{j}}(x_k)^\top d_k^{\mathcal{I}} + \left[\nabla f_{\hat{j}}(\xi_k) - \nabla f_{\hat{j}}(x_k)\right]^\top d_k^{\mathcal{I}} > \gamma\theta_k^{\mathcal{I}}.$$

Since $\hat{j} \in \mathcal{I}$ and the norm is non-negative, we have that $\theta_k^{\mathcal{I}} = \max_{j\in\mathcal{I}}\nabla f_j(x_k)^\top d_k^{\mathcal{I}} + \frac{1}{2}\|d_k^{\mathcal{I}}\|^2 \geq \nabla f_{\hat{j}}(x_k)^\top d_k^{\mathcal{I}}$ and, thus, $(1-\gamma)\theta_k^{\mathcal{I}} + \left[\nabla f_{\hat{j}}(\xi_k) - \nabla f_{\hat{j}}(x_k)\right]^\top d_k^{\mathcal{I}} > 0$. Using Equation (4.2), we obtain

$$-(1-\gamma)\bar{\varepsilon} + \left[\nabla f_{\hat{j}}(\xi_k) - \nabla f_{\hat{j}}(x_k)\right]^\top d_k^{\mathcal{I}} > 0.$$

By taking the limit for $k \to \infty, k \in \bar{K}$, recalling the continuity of $J_F(\cdot)$, the boundedness of $d_k^{\mathcal{I}}$ and that $\alpha_k \to 0$, we get that $-(1-\gamma)\bar{\varepsilon} > 0$. Since $1-\gamma > 0$ and $\bar{\varepsilon} > 0$, we get the contradiction. So, we prove that the limit point $\bar{x}$ of the sequence $\{x_k\}$ is Pareto-stationary w.r.t. $F_{\mathcal{I}}(\cdot)$. □

Finally, we prove that, when a stopping criterion based on the ε-Pareto-stationarity is considered, `FMOPG` is well defined, i.e., it terminates in a finite number of iterations.

Proposition 4.3. *Let Assumptions 3.2-4.1 hold with $\mathcal{I} \subseteq \{1, \dots, m\}$, the set X^0 and the point x_0. Let $\varepsilon > 0$. Then, the `FMOPG` algorithm finds in a finite number of steps a point x_k which is ε-Pareto-stationary w.r.t. $F_{\mathcal{I}}(\cdot)$.*

Proof. Since Assumptions 3.2-4.1 hold, Proposition 4.2 ensures that there exists a subsequence $K \subseteq \{1, 2, \dots\}$ such that

$$\lim_{\substack{k \to \infty \\ k \in K}} x_k = \bar{x}$$

and $\bar{x}$ is Pareto-stationary w.r.t. $F_{\mathcal{I}}(\cdot)$. From this point forward, the proof is similar to the one for Proposition 3.1. The noteworthy difference is that only the subset of objective functions indicated by $\mathcal{I}$ must be considered. □

4.3 Conclusions

In this chapter, we considered smooth multi-objective optimization problems subject to bound constraints. After a review of the existing literature, we listed and commented the benefits and drawbacks of the main state-of-the-art approaches designed to approximate the Pareto front of such problems, i.e., evolutionary algorithms and descent methods. We then focused on the design of a memetic algorithm, whose aim is to combine the good features of both the aforementioned approaches. We call this new method *Non-dominated Sorting Memetic Algorithm* (`NSMA`). In this procedure, we exploit the genetic operations of `NSGA-II`, which is the most popular genetic algorithm for the considered class of problems, and the tools typical of gradient-based descent methods, such as steepest descent directions and line searches. In particular, we employ a new descent method, called *Front Multi-Objective Projected Gradient* (`FMOPG`), which is a front-based variant of the original `MOPG` firstly introduced in [29]. For `FMOPG`, we proved properties of convergence to Pareto stationarity for the sequence of produced points.

Results of thorough computational experiments, in which we compared our method with main state-of-the-art algorithms (`NSGA-II` included), are reported in Section 9.2. These results show that `NSMA` can consistently outperform its competitors in terms of popular metrics for multi-objective optimization.

Chapter 5

A Limited Memory Quasi-Newton Approach for Multi-Objective Optimization

In this chapter [1], we deal with the class of unconstrained multi-objective optimization problems, i.e.,

$$\min_{x \in \mathbb{R}^n} \; F(x) = (f_1(x), \ldots, f_m(x))^\top. \tag{5.1}$$

Quasi-Newton methods are among the most popular algorithms for the unconstrained setting in scalar optimization. Based on a quadratic model of the objective function, they do not require the calculation of the second derivatives in order to find the search direction: the real Hessian is replaced by an approximation matrix, which is updated at each iteration considering the new generated solution and the previous one. The most famous update formula for the approximation matrix is the BFGS one, which is named after Broyden, Fletcher, Goldfarb and Shanno [5]. In the multi-objective setting, Quasi-Newton methods were proposed, for instance, in [1, 89–91].

Among the factors contributing to the success of Quasi-Newton methods in scalar optimization, the possibility of defining limited-memory variants of these approaches certainly stands out. The approximate Hessian matrix can in fact be roughly recovered only using a finite number M of previously

[1] Part of the content of this chapter has been published as "A limited memory Quasi-Newton approach for multi-objective optimization" in *Computational Optimization and Applications, 2023* [60].

generated solutions. In this way, its management in memory, which could be extremely inefficient and time-consuming, is avoided. In particular, the L-BFGS algorithm, firstly designed in [83], has managed over the years to achieve state-of-the-art performance in most settings, even with relatively small values for M.

This work concerns, to the best of our knowledge, the first attempt in the literature to define a multi-objective limited memory Quasi-Newton method. The key elements that characterize the proposed approach are the following.

- A shared approximation of the Hessian matrices is employed to compute the search direction.
- The Hessian matrix approximation only requires information related to the most recent iterations to be computed.
- Equipped with a Wolfe type line search, the method is in general well defined; moreover, in the strongly convex case, it is shown to possess R-linear global convergence properties to Pareto optimality.

5.1 The Algorithm

We report the algorithmic scheme of our new Limited Memory Quasi-Newton approach for MOO in Algorithm 5.1.

In the proposed approach, we use a single positive definite matrix H^k at each iteration k. In Section 5.1.2, we introduce the update formula for H^k, which is slightly different w.r.t. the one introduced in [1] and reported in Section 2.2.2. As in L-BFGS for scalar optimization, we maintain only a finite number M of vectors pairs $\{(s_i, u_i)\}$ in memory: the oldest one is discarded each time a new vectors pair is calculated. These pairs are used in a two-loop recursive procedure to efficiently carry out the matrix multiplication $\mathcal{R}^k = H^k J_F(x_k)^\top$ (Section 5.1.1). This procedure is essentially an extension for MOO of the one used in L-BFGS [84]. The matrix $\mathcal{R}^k$ is then used in Problem (5.2) at Step 4 of the algorithm: the latter is simply derived from Problem (2.9) substituting $H^k J_F(x_k)^\top$ with $\mathcal{R}^k$. We denote by $\theta^{LM}(x_k)$ the optimal value of Problem (5.2) at x_k. Moreover, we respectively denote by $\lambda^{LM}(x_k)$ (Line 5) and $v^{LM}(x_k)$ (Line 6) the Lagrange multipliers vector and the direction corresponding to $\theta^{LM}(x_k)$. Note that (2.6) is valid in this context too. Finally, in Line 7, a Wolfe line search is carried out to find a step size α_k along the direction $v^{LM}(x_k)$, satisfying the Wolfe conditions for

Algorithm 5.1: Limited Memory Quasi-Newton Method

Input: $F : \mathbb{R}^n \to \mathbb{R}^m$, $x_0 \in \mathbb{R}^n$, $\gamma \in (0, 1/2)$, $\sigma \in (\gamma, 1)$, $H^0 \succ 0$, $M \in \mathbb{N}^+$.

for $k = 0, 1, 2, \ldots$ **do**

Estimate $\mathcal{R}^k = H^k J_F(x_k)^\top \in \mathbb{R}^{n \times m}$ (two-loop recursive procedure)

Compute

$$\begin{aligned} \theta^{LM}(x_k) = \max_{\lambda \in \mathbb{R}^m} \; & -\frac{1}{2}\lambda^\top J_F(x_k)\mathcal{R}^k\lambda \\ \text{s.t.} \; & \sum_{j=1}^{m} \lambda_j = 1, \qquad \lambda \geq \mathbf{0}_m \end{aligned} \tag{5.2}$$

Let $\lambda^{LM}(x_k) = \left(\lambda_1^{LM}(x_k), \ldots, \lambda_m^{LM}(x_k)\right)^\top$ be the Lagrange multipliers vector related to $\theta^{LM}(x_k)$

Let $v^{LM}(x_k) = -\mathcal{R}^k\lambda^{LM}(x_k)$

Choose $\alpha_k > 0$ (trying first $\alpha_k = 1$) such that

$$F\left(x_k + \alpha_k v^{LM}(x_k)\right) \leq F(x_k) + \mathbf{1}_m \gamma \alpha_k D\left(x_k, v^{LM}(x_k)\right) \tag{5.3}$$

$$D\left(x_k + \alpha_k v^{LM}(x_k), v^{LM}(x_k)\right) \geq \sigma D\left(x_k, v^{LM}(x_k)\right) \tag{5.4}$$

Let $x_{k+1} = x_k + \alpha_k v^{LM}(x_k)$

if $k \geq M$ **then**

Discard vectors pair (s_{k-M}, u_{k-M}) from storage

Compute and save

$$s_k = x_{k+1} - x_k \tag{5.5}$$

$$u_k = \sum_{j=1}^{m} \lambda_j^{LM}(x_k)\left[\nabla f_j(x_{k+1}) - \nabla f_j(x_k)\right] \tag{5.6}$$

return x_k

MOO (Section 5.1.3). The function $D(\cdot, \cdot)$ present in the conditions, as well as in the rest of the chapter, is defined as in (2.4).

In the following, we deeply analyze the various aspects of Algorithm 5.1.

5.1.1 Two-Loop Recursive Procedure for MOO

In L-BFGS, one of the most relevant features is the two-loop recursive procedure which, at any iteration k, given the vectors pairs saved in memory, allows to efficiently compute the product $H^k \nabla f(x_k)$, where $f(\cdot)$ indicates the objective function [84]. We remind, indeed, that in scalar optimization the negative of this product identifies the Quasi-Newton descent direction: $d(x_k) = -H^k \nabla f(x_k)$. Using this procedure, we do not need to store the matrix H in memory. This property could be crucial when high dimensional problems are considered: in these cases, maintaining and updating the matrix H, which is dense in general, could be extremely inefficient. Here, we propose an extension of this procedure for MOO: the algorithmic scheme is reported in Algorithm 5.2.

Algorithm 5.2: Two-Loop Recursive Procedure

Input: $k \in \mathbb{N}$, $M \in \mathbb{N}^+$, $\{(s_i, u_i) \mid i \in [\max\{0, k-M\}, k-1]\}$, $J_F(x_k) \in \mathbb{R}^{m \times n}$, $H^0 \succ 0$.
$q = J_F(x_k)^\top$
if $k = 0$ **then**
    $\mathcal{R}^0 = H^0 q$
else
    for $i = k-1, \ldots, \max\{0, k-M\}$ **do**
        $\alpha_i = \rho^i q^\top s_i$
        $q = q - u_i \alpha_i^\top$
    $\mathcal{R}^k = H^0 q$
    for $i = \max\{0, k-M\}, \ldots, k-1$ **do**
        $\beta_i = \rho^i \left(\mathcal{R}^k\right)^\top u_i$
        $\mathcal{R}^k = \mathcal{R}^k + s_i \left(\alpha_i - \beta_i\right)^\top$
return $\mathcal{R}^k$

With respect to the scalar optimization case, this procedure computes the product $H^k J_F(x_k)^\top$. The same result could be obtained repeating m times the procedure for scalar optimization to find $H^k \nabla f_j(x_k)$ for all $j \in \{1, \ldots, m\}$. In both cases, $m\,(4Mn + n)$ multiplications are required.

However, Algorithm 5.2 allows to exploit the optimized operations of software libraries for vector calculus.

The employment of Algorithm 5.2 is possible thanks to some properties of (2.11). Indeed, the latter can be re-written in the following form [84]:

$$\begin{aligned} H^k = &\left[\left(V^{k-1}\right)^\top \dots \left(V^{k-M}\right)^\top\right] H^{k-M} \left[V^{k-M} \dots V^{k-1}\right] \\ &+ \rho^{k-M} \left[\left(V^{k-1}\right)^\top \dots \left(V^{k-M+1}\right)^\top\right] s_{k-M} s_{k-M}^\top \left[V^{k-M+1} \dots V^{k-1}\right] \\ &+ \rho^{k-M+1} \left[\left(V^{k-1}\right)^\top \dots \left(V^{k-M+2}\right)^\top\right] s_{k-M+1} s_{k-M+1}^\top \left[V^{k-M+2} \dots V^{k-1}\right] \\ &+ \dots \\ &+ \rho^{k-1} s_{k-1} s_{k-1}^\top, \end{aligned}$$

where $V^i = I_n - \rho^i u_i s_i^\top$. As in L-BFGS, the *exact* matrix H^{k-M} is substituted by a suitable sparse positive definite matrix H^0. From this last equation, the two-loop recursive procedure to compute the product $H^k J_F(x_k)^\top$ is derived. We refer the reader to [84] for more details.

5.1.2 Definition of H

In the proposed approach, we use a single positive definite matrix H. As in [1] the update formula (2.11) is used. However, taking inspiration from (3.2), we use a different definition of ρ^k:

$$\rho^k = \begin{cases} 1/\left(s_k^\top u_k\right) & \text{if } s_k^\top u_k > 0, \\ 1/\left\{\sum_{j=1}^m \lambda_j^{LM}(x_k) \left[D\left(x_{k+1}, s_k\right) - \nabla f_j\left(x_k\right)^\top s_k\right]\right\} & \text{otherwise.} \end{cases} \tag{5.7}$$

As in [90], we carry out a line search to find a step size satisfying the Wolfe conditions for MOO (Section 5.1.3). However, recalling the reasoning in Remark 3.2, in order to ensure that $H^{k+1} \succ 0$, we force through (5.7) ρ^k to be positive even when $s_k^\top u_k \leq 0$. We formalize this statement in the following proposition.

Proposition 5.1. *Considering a generic iteration k of Algorithm 5.1, let $x_k \in \mathbb{R}^n$, $v^{LM}(x_k) \in \mathbb{R}^n$ be a direction such that $D\left(x_k, v^{LM}(x_k)\right) < 0$, $\alpha_k > 0$ be a step size along $v^{LM}(x_k)$ and $\lambda^{LM}(x_k)$ be the Lagrange multipliers vector obtained solving Problem (5.2). If ρ^k is updated by (5.7), then ρ^k is positive.*

Proof. See Appendix C. □

Remark 5.1. In the single objective case, the update formula (2.11) for H^k coincides with the classical BFGS rule. Indeed, it is sufficient to realize that, since $\lambda^{LM}(x_k)$ lies in the unit simplex by (2.6), then $u_k = y^k$. Moreover, the same reasoning can be applied with (5.7) to get that $\rho^k = 1/\left(s_k^\top y^k\right)$. Hence, the two-loop recursive procedure reduces to that of L-BFGS. In turn, the overall Algorithm 5.1 is nothing but L-BFGS, since $d(x_k) = -H^k \nabla f(x_k)$ and Wolfe conditions are imposed by the line search.

Remark 5.2. The procedure in Algorithm 5.2 cannot be used if we consider an approximation matrix for each objective function, as in Problem (2.3) with $M_j(\cdot) = B_j$ for all $j \in \{1, \ldots, m\}$ (see also Table 2.1). In such case, both in the primal and in the dual problem (2.5) the matrices are tied to the problem variables; for example, when solving (2.5), the product $[\sum_{j=1}^m \lambda_j B_j]^{-1} J_F(\bar{x})^\top$ would be recomputed any time a different solution λ is considered. The use of a single positive definite matrix prevents this issue: matrix multiplication $HJ_F(\bar{x})^\top$ can be computed only once, before solving subproblem (2.9), making it possible to exploit the efficiency of the two-loop recursive procedure.

5.1.3 Wolfe Line Search

In this section, we introduce a simple line search scheme to find a step size α along a given direction d_k satisfying the Wolfe conditions:

$$F(x_k + \alpha d_k) \leq F(x_k) + \mathbf{1}_m \gamma \alpha D(x_k, d_k), \tag{5.8}$$

$$D(x_k + \alpha d_k, d_k) \geq \sigma D(x_k, d_k). \tag{5.9}$$

Before proceeding, we consider that Assumption 3.1 with $\Omega = \mathbb{R}^n$ holds. Then, we prove that there exists an interval of values satisfying the Wolfe conditions. Note that an analogous result has been obtained in [71,72] under the different assumptions also reported in Section 3.2.2 of this manuscript.

Proposition 5.2. *Let Assumption 3.1 hold with $\Omega = \mathbb{R}^n$. Let $x_k \in \mathbb{R}^n$ and assume that $d_k \in \mathbb{R}^n$ is a direction such that $D(x_k, d_k) < 0$, $\gamma \in (0, 1/2)$ and $\sigma \in (\gamma, 1)$. Then, there exists an interval of values $[\alpha_l, \alpha_u]$, with $0 < \alpha_l < \alpha_u$, such that for all $\alpha \in [\alpha_l, \alpha_u]$ Equations (5.8) and (5.9) hold.*

Proof. See Appendix C. □

After proving the existence of an interval of values satisfying the Wolfe conditions, we report the algorithmic scheme of the considered line search.

Algorithm 5.3: Wolfe Line Search

```
1  Input: F : ℝ^n → ℝ^m, x_k ∈ ℝ^n, d_k ∈ ℝ^n, γ ∈ (0, 1/2), σ ∈ (γ, 1).
2  α_l^0 = 0, α_u^0 = ∞, α^0 = 1
3  for t = 0, 1, 2, ... do
4      if ∃j s.t. f_j(x_k + α^t d_k) > f_j(x_k) + γα^t D(x_k, d_k) then
5          α_u^{t+1} = α^t
6          α_l^{t+1} = α_l^t
7      else
8          α_u^{t+1} = α_u^t
9          if D(x_k + α^t d_k, d_k) < σD(x_k, d_k) then
10             α_l^{t+1} = α^t
11         else
12             return α^t
13     Choose α^{t+1} ∈ (α_l^{t+1}, α_u^{t+1})
```

Starting from $\alpha_l^0 = 0, \alpha_u^0 = \infty$, the core idea of the line search is that of reducing the interval $[\alpha_l^t, \alpha_u^t]$ until a valid step size α^t is found. At the beginning of the for-loop, the Wolfe sufficient decrease condition (5.8) is checked. If it is not satisfied by α^t, we update α_u^t and we maintain the same value for α_l^t (Lines 5 and 6). Otherwise, α_u^t is not updated (Line 8) and we check if the Wolfe curvature condition (5.9) is satisfied by α^t: if it is, both Wolfe conditions are satisfied and, then, the current step size value is returned; else α_l^t is updated according to Line 10. After updating α_u^t or α_l^t, a new value for the step size α^t is chosen in the interval (α_l^t, α_u^t) and the process is repeated.

In the next lemma, we state some properties related to the interval upper and lower bounds α_u^t and α_l^t.

Lemma 5.1. *Consider a generic iteration t of Algorithm 5.3. Let $x_k \in \mathbb{R}^n$ and d_k be a direction such that $D(x_k, d_k) < 0$. Then, we have the following properties:*

1. *if $\alpha_u^t < \infty$, then*

$$\exists\, j\big(\alpha_u^t\big) \;\; s.t. \;\; f_{j(\alpha_u^t)}(x_k + \alpha_u^t d_k) > f_{j(\alpha_u^t)}(x_k) + \gamma \alpha_u^t D(x_k, d_k);$$

2. α_l^t *is such that*

$$F(x_k + \alpha_l^t d_k) \leq F(x_k) + \mathbf{1}_m \gamma \alpha_l^t D(x_k, d_k),$$
$$D(x_k + \alpha_l^t d_k, d_k) < \sigma D(x_k, d_k).$$

Proof. See Appendix C. □

In the following proposition, we state that the proposed line search is well defined, i.e., it terminates after a finite number of iterations returning a step size satisfying the Wolfe conditions.

Proposition 5.3. *Let Assumption 3.1 hold with* $\Omega = \mathbb{R}^n$, $\delta \in [1/2, 1)$, $\eta > 1$ *and let* $\{\alpha_l^t, \alpha_u^t, \alpha^t\}$ *be the sequence generated by Algorithm 5.3. Assume that:*

1. $d_k \in \mathbb{R}^n$ *is a descent direction for* $F(\cdot)$ *at* $x_k \in \mathbb{R}^n$*;*
2. *for all* $t > 0$*, the step size* α^t *is chosen so that*
   a. *if* $\alpha_u^t = \infty$,
   $$\alpha^t \geq \eta \max\left\{\alpha_l^t, \alpha^0\right\},$$
   b. *if* $\alpha_u^t < \infty$,
   $$\max\left\{\left(\alpha^t - \alpha_l^t\right), \left(\alpha_u^t - \alpha^t\right)\right\} \leq \delta\left(\alpha_u^t - \alpha_l^t\right).$$

Then Algorithm 5.3 is well defined, i.e., it stops after a finite number of iterations returning a step size $\hat{\alpha}$ *satisfying the Wolfe conditions for MOO.*

Proof. See Appendix C. □

Remark 5.3. To the best of our knowledge, the first Wolfe line search for MOO was proposed in [72]. Our line search is just a simpler algorithm that is guaranteed to produce a point satisfying the Wolfe conditions. In fact, we think that not using an inner solver, as done in [72], could be a performance disadvantage and, in addition, smarter strategies to set the trial step size may be integrated. We decided not to compare the two line searches, since finding new efficient methodologies to find the step size is not the focus of our work. Moreover, we are confident that the experimental results of Section 9.3 would be similar regardless the employed Wolfe line search.

5.2 Convergence Analysis

In this section, we show the convergence properties of our Limited Memory Quasi-Newton approach. Before proceeding, similarly to what is done in [65], we need to make some assumptions about the objective function $F(\cdot)$ and the initial approximation matrix H^0.

Assumption 5.1. We assume that:

- $F(\cdot)$ is twice continuously differentiable;
- the set $\mathcal{L}_F\left(F\left(x_0\right)\right) = \{x \in \mathbb{R}^n \mid F\left(x\right) \leq F\left(x_0\right)\}$ is convex;
- $\exists a, b \in \mathbb{R}^+$ such that, for all $j \in \{1, \ldots, m\}$,

$$a\left\|z\right\|^2 \leq z^\top \nabla^2 f_j(x) z \leq b\left\|z\right\|^2, \qquad \forall z \in \mathbb{R}^n, \forall x \in \mathcal{L}_F\left(F(x_0)\right).$$

Assumption 5.2. The matrix H^0 is chosen such that the norms $\left\|H^0\right\|$ and $\left\|B^0\right\|$ are bounded.

Remark 5.4. Assumption 5.1 implies Assumption 3.1 with $\Omega = \mathbb{R}^n$. Indeed, $f_j\left(\cdot\right)$ is strongly convex for all $j \in \{1, \ldots, m\}$ and, thus, has all the level sets bounded. Also, by Assumption 3.1, we have that Propositions 5.2 and 5.3 concerning the line search remain valid.

Remark 5.5. By Assumption 5.1, we have $s_k^\top y_j^k > 0$ for any k and for all $j \in \{1, \ldots, m\}$. Then, considering (5.6) and since $\lambda^{LM}\left(x_k\right)$ satisfies (2.6), we have that $s_k^\top u_k > 0$. Then, according to (5.7), $\rho^k = 1/\left(s_k^\top u_k\right)$ and, thus, we update B^k and H^k using (2.10) and (2.11), respectively.

In order to carry out the theoretical analysis, we take as reference Algorithm 5.4, which is mathematically equivalent to Algorithm 5.1 but makes it more explicit how the approximation of H^k is computed, i.e., applying M times the update rule (2.11) starting from H^0. In the remainder of the section, we will consider the approximation matrix B^k for the sake of clarity; the results are obviously the same if we consider the matrix H^k. Finally, note that Algorithm 5.4 is only used in this section, since, unlike Algorithm 5.1, it requires to store the entire matrix in memory.

Algorithm 5.4: Limited Memory Quasi-Newton Method

Input: $F : \mathbb{R}^n \to \mathbb{R}^m$, $x_0 \in \mathbb{R}^n$, $\gamma \in (0, 1/2)$, $\sigma \in (\gamma, 1)$, $B^0 \succ 0$, $M \in \mathbb{N}^+$.
Let $h = 0$
for $k = 0, 1, 2, \ldots$ **do**
  Compute $\theta^{LM}(x_k)$ and $\lambda^{LM}(x_k)$ solving Problem (2.9)
  Let $v^{LM}(x_k) = -\left(B^k\right)^{-1} J_F\left(x_k\right)^\top \lambda^{LM}(x_k)$
  Choose $\alpha_k > 0$ s.t. (5.3) and (5.4) hold
  Let $x_{k+1} = x_k + \alpha_k v^{LM}(x_k)$
  if $k \geq M$ **then**
    Discard vectors pair (s_{k-M}, u_{k-M}) from storage
    Set $h = k - M + 1$
  Compute and save (s_k, u_k) according to (5.5)-(5.6)
  Let $B^k_{(0)} = B^0$
  for $l = 0, \ldots, \min\{k, M-1\}$ **do**
    Set

$$B^k_{(l+1)} = B^k_{(l)} - \frac{B^k_{(l)} s_{l+h} s_{l+h}^\top B^k_{(l)}}{s_{l+h}^\top B^k_{(l)} s_{l+h}} + \frac{u_{l+h} u_{l+h}^\top}{s_{l+h}^\top u_{l+h}} \tag{5.10}$$

  Let $B^{k+1} = B^k_{(\min\{k,M-1\}+1)}$
return x_k

For the theoretical analysis, we also need to introduce the formula for the trace and the determinant of the matrix B^{k+1}:

$$\mathrm{Tr}(B^{k+1}) = \mathrm{Tr}(B^k) - \frac{\left\|B^k s_k\right\|^2}{s_k^\top B^k s_k} + \frac{\|u_k\|^2}{s_k^\top u_k}, \tag{5.11}$$

$$\det(B^{k+1}) = \det(B^k) \frac{s_k^\top u_k}{s_k^\top B^k s_k}. \tag{5.12}$$

Note that these expressions hold when (2.10) is used to update the matrix B^k, which is always the case here by Assumption 5.1. We also introduce some basic notation that will be useful in the following analysis.

Additional Notation We will denote by $\Omega\left(B^k\right)$ the eigenvalues set of the matrix B^k; by $\omega_m\left(B^k\right)$ and $\omega_M\left(B^k\right)$ we indicate the minimum and the maximum eigenvalue, respectively; we refer by β^k to the angle between the vectors s_k and $B^k s_k$. Concerning β^k, we also recall the formula of the cosine:

$$\cos\beta^k = \frac{s_k^\top B^k s_k}{\|s_k\| \, \|B^k s_k\|}. \tag{5.13}$$

We are now able to begin the convergence analysis with three technical lemmas.

Lemma 5.2. *Let Assumption 5.1 hold and consider the sequences $\{x_k\}$ and $\left\{v^{LM}\left(x_k\right)\right\}$ generated by Algorithm 5.4. Then,*

$$\sum_{k\geq 0} \frac{D\left(x_k, v^{LM}(x_k)\right)^2}{\|v^{LM}(x_k)\|^2} < \infty.$$

Proof. The result follows as in Proposition 3.3 in [90], as the assumptions made in the latter are trivially implied by Assumption 5.1. □

Lemma 5.3. *Consider Assumption 5.1 and let $\{x_k\}$ be the sequence generated by Algorithm 5.4. Then, for all $k \geq 0$, we have that*

$$D\left(x_k, v^{LM}(x_k)\right) \leq -\frac{\cos\beta^k}{2} \left\|v^{LM}(x_k)\right\| \left\|v^{SD}(x_k)\right\|,$$

where $v^{SD}(x_k)$ is the steepest common descent direction calculated at x_k (see Table 2.1).

Proof. The proof is analogous to the one of Lemma 4.2 in [90], taking into account that we have a single approximation matrix B^k. □

Lemma 5.4. *Let Assumptions 5.1 and 5.2 hold. Moreover, let $\{x_k\}$ be the sequence generated by Algorithm 5.4. Then, there exists a constant $\delta > 0$ such that, for all $k \geq 0$, we have that*

$$\cos\beta^k \geq \delta.$$

Proof. See Appendix C. □

In the next proposition, we state that the sequence of points produced by Algorithm 5.4 converges to a Pareto optimal point.

Proposition 5.4. *Let Assumptions 5.1 and 5.2 hold. Assume that* $\{x_k\}$ *is the sequence generated by Algorithm 5.4. Then,* $\{x_k\}$ *converges to a Pareto optimal point* $x^\star$ *for Problem* (5.1).

Proof. By Lemmas 5.3 and 5.4, we know that there exists a constant $\delta > 0$ such that, for all $k \geq 0$,

$$\begin{aligned} D\left(x_k, v^{LM}(x_k)\right) &\leq -\frac{\cos\beta^k}{2}\left\|v^{LM}(x_k)\right\|\left\|v^{SD}(x_k)\right\| \\ &\leq -\frac{\delta}{2}\left\|v^{LM}(x_k)\right\|\left\|v^{SD}(x_k)\right\|. \end{aligned}$$

Considering this last result and Lemma 5.2, we obtain that

$$\infty > \sum_{k\geq 0}\frac{D\left(x_k, v^{LM}(x_k)\right)^2}{\left\|v^{LM}(x_k)\right\|^2} \geq \sum_{k\geq 0}\frac{\delta^2}{4}\left\|v^{SD}(x_k)\right\|^2,$$

and, thus,

$$\lim_{k\to\infty} v^{SD}(x_k) = \mathbf{0}_n. \tag{5.14}$$

By (5.3), we know that, for all $k \geq 0$, $x_k \in \mathcal{L}_F\left(F\left(x_0\right)\right)$. Since $\mathcal{L}_F\left(F\left(x_0\right)\right)$ is compact (Remark 5.4), there exists a subsequence $K \subseteq \{0, 1, \ldots\}$ such that

$$\lim_{\substack{k\to\infty \\ k\in K}} x_k = x^\star. \tag{5.15}$$

Recalling Equation (5.14), Lemmas 2.4 and 2.2 with $\Omega = \mathbb{R}^n$ and $M_j(\cdot) = I_n$ for all $j \in \{1, \ldots, m\}$, we have that $v^{SD}\left(x^\star\right) = \mathbf{0}_n$ and, thus, $x^\star$ is Pareto-stationary for Problem (5.1). Therefore, by Lemma 2.1 with $\Omega = \mathbb{R}^n$ and Assumption 5.1, we conclude that $x^\star$ is Pareto optimal.

Now, let us assume, by contradiction, that there exists another subsequence $\tilde{K} \subseteq \{0, 1, \ldots\}$ such that

$$\lim_{\substack{k\to\infty \\ k\in \tilde{K}}} x_k = \tilde{x}, \tag{5.16}$$

with $\tilde{x} \neq x^\star$.

We prove that $F\left(\tilde{x}\right) \neq F\left(x^\star\right)$. If it were false, since by Assumption 5.1 F is strongly convex and $\mathcal{L}_F\left(F\left(x_0\right)\right)$ is convex, for all $t \in (0, 1)$, we would get that $F\left(t\tilde{x} + (1-t)\,x^\star\right) < tF\left(\tilde{x}\right) + (1-t)\,F\left(x^\star\right) = F\left(x^\star\right)$. But, in this case, we would contradict the fact that $x^\star$ is Pareto optimal.

Then, given that $x^\star$ is Pareto optimal and that $F(\tilde{x}) \neq F(x^\star)$, $\exists \tilde{j} \in \{1, \ldots, m\}$ such that $f_{\tilde{j}}(x^\star) < f_{\tilde{j}}(\tilde{x})$. Now, recalling (5.15) and (5.16), there exist $k \in K$ and $\tilde{k} \in \tilde{K}$ such that $k < \tilde{k}$ and $f_{\tilde{j}}(x_k) < f_{\tilde{j}}(x_{\tilde{k}})$. But, since (5.3) holds at each iteration of Algorithm 5.4, we implicitly have that the sequence $\{f_j(x_k)\}$ is decreasing, for all $j \in \{1, \ldots, m\}$. Thus, we get a contradiction and we conclude that

$$\lim_{k \to \infty} x_k = x^\star,$$

with $x^\star$ being Pareto optimal. □

In the rest of the section, we discuss the convergence rate of Algorithm 5.4. We first have to provide a technical result.

Lemma 5.5. *Let Assumptions 5.1 and 5.2 hold. Moreover, let $\{x_k\}$ be the sequence generated by Algorithm 5.4 and $x^\star$ be the Pareto optimal point to which the sequence converges. Then, for all $k \geq 0$,*

- $\|x_k - x^\star\| \leq \frac{2}{a} \left\| v^{SD}(x_k) \right\|$,
- $\|s_k\| \geq \frac{(1-\sigma)}{2b} \cos\beta^k \left\| v^{SD}(x_k) \right\|$,

where $v^{SD}(x_k)$ is the steepest common descent direction calculated at x_k (see Table 2.1).

Proof. The proof is analogous to the one of Lemma 4.4 in [90], recalling that here a single approximation matrix B^k is considered. □

We are now ready to prove that the sequence of points generated by Algorithm 5.4 R-linearly converges to Pareto optimality.

Proposition 5.5. *Let Assumptions 5.1 and 5.2 hold. Furthermore, let $\{x_k\}$ be the sequence generated by Algorithm 5.4 and $x^\star$ be the Pareto optimal limit point of the sequence. Then, $\{x_k\}$ R-linearly converges to $x^\star$. In addition, we have that*

$$\sum_{k \geq 0} \|x_k - x^\star\| < \infty. \tag{5.17}$$

Proof. We first introduce the function $f^\star : \mathbb{R}^n \to \mathbb{R}$, defined as

$$f^\star(x) = \sum_{j=1}^{m} \lambda_j^{SD}(x^\star) f_j(x), \tag{5.18}$$

where $\lambda^{SD}(x^\star)$ is the multipliers vector associated with the steepest common descent direction at $x^\star$. Recalling Lemmas 2.1 and 2.2 with $\Omega = \mathbb{R}^n$ and $M_j(\cdot) = I_n$ for all $j \in \{1, \ldots, m\}$, that $x^\star$ is Pareto optimal and that (2.7) holds for $v^{SD}(x^\star)$, we have that

$$\nabla f^\star(x^\star) = \sum_{j=1}^{m} \lambda_j^{SD}(x^\star) \nabla f_j(x^\star) = -v^{SD}(x^\star) = \mathbf{0}_n. \tag{5.19}$$

Now, for all $k \geq 0$ and $j \in \{1, \ldots, m\}$, by Assumption 5.1 and using Taylor's theorem, we get $\frac{a}{2}\|x_k - x^\star\|^2 \leq f_j(x_k) - f_j(x^\star) - \nabla f_j(x^\star)^\top (x_k - x^\star) \leq \frac{b}{2}\|x_k - x^\star\|^2$. Multiplying this result by $\lambda_j^{SD}(x^\star)$, summing over $j \in \{1, \ldots, m\}$, recalling (2.6), which is valid for $\lambda^{SD}(x^\star)$, and (5.19), we obtain that

$$\frac{a}{2}\|x_k - x^\star\|^2 \leq f^\star(x_k) - f^\star(x^\star) \leq \frac{b}{2}\|x_k - x^\star\|^2. \tag{5.20}$$

Given Lemma 5.5, from the right-hand side of the last result we get

$$f^\star(x_k) - f^\star(x^\star) \leq \frac{2b}{a^2}\left\|v^{SD}(x_k)\right\|^2. \tag{5.21}$$

On the other side, (2.6), (5.3) and (5.18) imply that, for all $k \geq 0$, $f^\star(x_{k+1}) \leq f^\star(x_k) + \gamma\alpha_k D\left(x_k, v^{LM}(x_k)\right)$ which, by subtracting the term $f^\star(x^\star)$ in both sides and taking into account Lemmas 5.3 and 5.5, changes into

$$\begin{aligned} f^\star(x_{k+1}) - f^\star(x^\star) &\leq f^\star(x_k) - f^\star(x^\star) - \frac{\gamma\cos\beta^k}{2}\|s_k\|\left\|v^{SD}(x_k)\right\| \\ &\leq f^\star(x_k) - f^\star(x^\star) - \frac{\gamma(1-\sigma)\cos^2\beta^k}{4b}\left\|v^{SD}(x_k)\right\|^2. \end{aligned}$$

Joining this last result and (5.21), we obtain that

$$f^\star(x_{k+1}) - f^\star(x^\star) \leq r_k\left(f^\star(x_k) - f^\star(x^\star)\right), \tag{5.22}$$

with $r_k = 1 - \frac{\gamma(1-\sigma)a^2\cos^2\beta^k}{8b^2}$, for all $k \geq 0$. It is easy to see that, by the definitions of γ and σ, Assumption 5.1 and Lemma 5.4, $r_k \in (0, 1)$. In addition, by Lemma 5.4, we also have that there exists a constant $\delta > 0$ such that, for all $k \geq 0$, $r_k \leq 1 - \frac{\gamma(1-\sigma)a^2\delta^2}{8b^2} = \bar{r} < 1$. Then, recursively applying Equation (5.22) and taking into account that, combining (2.6), (5.3) and

(5.18), $f^\star(x_0) - f^\star(x^\star) > 0$, we get

$$\begin{aligned} f^\star(x_{k+1}) - f^\star(x^\star) &\leq \left[\prod_{l=0}^{k} r_l\right] (f^\star(x_0) - f^\star(x^\star)) \\ &\leq \left[\prod_{l=0}^{k} \bar{r}\right] (f^\star(x_0) - f^\star(x^\star)) \\ &= \bar{r}^{k+1} (f^\star(x_0) - f^\star(x^\star)) . \end{aligned}$$

Considering this last result and the left-hand side of (5.20), we obtain that

$$\|x_{k+1} - x^\star\| \leq \left(\bar{r}^{k+1}\right)^{1/2} \left[\frac{2}{a} (f^\star(x_0) - f^\star(x^\star))\right]^{1/2},$$

and, thus, the sequence $\{x_k\}$ R-linearly converges to $x^\star$.

Summing the last result for all $k \geq 0$ and recalling that $\bar{r} < 1$, we get that (5.17) holds. □

5.3 Conclusions

In this chapter we proposed a new limited memory Quasi-Newton algorithm for unconstrained multi-objective optimization. To the best of our knowledge, it is the first attempt to define such an approach for MOO. As in [1], we use a single approximation matrix, contrarily to what is done in the other Quasi-Newton approaches. The idea of a single matrix, whose update formula is slightly modified from the one used in the scalar case, allowed us to extend the L-BFGS two-loop recursive procedure to multi-objective optimization: the Hessian matrix approximation does not need to be maintained and managed in memory, but it is computed using a finite number M of previously generated solutions. This feature is potentially crucial, especially when the approximation matrix is dense and/or high dimensional problems are handled. For the proposed approach, under assumptions similar to the ones made for L-BFGS in the strongly convex scalar case, we stated properties of R-linear convergence to the Pareto optimality of the produced sequence of points.

The results of thorough computational experiments, provided in Section 9.3, show that the new limited memory algorithm consistently outperforms the state-of-the-art Newton and Quasi-Newton methods for MOO. More-

over, the substantial benefits of using the proposed algorithm as local search procedure within a global optimization framework are highlighted.

Chapter 6

Improved Front Steepest Descent for Multi-objective Optimization

In this chapter [1], we are interested in optimization problems of the form

$$\min_{x \in \mathbb{R}^n} \; F(x) = (f_1(x), \ldots, f_m(x))^\top \tag{6.1}$$

and we focus on the *Front Steepest Descent* (`FSD`) algorithm proposed in [23]. For a brief description of the method, the reader is also referred to Section 3.3 of this manuscript. `FSD` was shown to be far superior than a simple multi-start version of the original (single-point) `MOSD` algorithm (Section 3.1.1). Yet, we argue that `FSD`, as defined in [23], has limited exploration capabilities and it is quite frequently unable to span large portions of the Pareto front.

We thus propose small but crucial modifications to the algorithm, that allow to turn it tremendously effective at spanning the entire Pareto front, regardless of the starting set of points. We then show that the proposed approach still enjoys the nice convergence guarantees of the original `FSD`.

[1] Part of the content of this chapter has been published as "Improved front steepest descent for multi-objective optimization" in *Operations Research Letters, 2023* [59].

6.1 FSD May Not Span the Pareto Front

The FSD algorithm constitutes, in practice, a significant improvement w.r.t. the simple multi-start steepest descent strategy for multi-objective optimization. However, in experimental settings, it is not uncommon to observe situations where FSD is unable to retrieve large portions of the Pareto front.

Here, we highlight this shortcoming and argue that it is the direct result of algorithmic design. In particular, the first condition at Step 6 of Algorithm 3.4 makes the outcome of the algorithm very strongly dependent on the starting point(s).

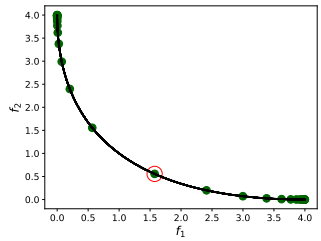
When a point x_c is considered for exploration in Algorithm 3.4, a partial descent direction obtained according to the subset of objectives $\mathcal{I} \subseteq \{1, \ldots, m\}$ is only considered if x_c is nondominated within X^k w.r.t. $F_{\mathcal{I}}(\cdot)$; in other words, there is no $y \in X^k$ such that $F_{\mathcal{I}}(y) \lneqq F_{\mathcal{I}}(x_c)$. This condition was required by the authors of [23] in order to establish finite termination properties for the line search (Algorithm 3.3).

Unfortunately, that same condition results in a limited fraction of points in X^k to be used for starting a partial descent search. This fact can be visualized, with very extreme outcomes, in the bi-objective case; indeed, when $m = 2$, for each of the two proper subsets of indices, $\mathcal{I}_1 = \{1\}$ and $\mathcal{I}_2 = \{2\}$, there is only one point that satisfies the (partial) nondominance condition: $x_{\mathcal{I}_1} = \arg\min_{x \in X^k} f_1(x)$ and $x_{\mathcal{I}_2} = \arg\min_{x \in X^k} f_2(x)$.

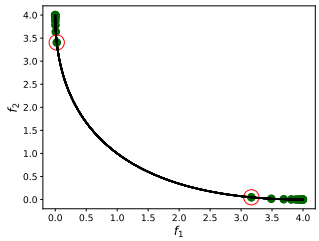
Thus, partial descent is only carried out starting from the two current extreme points in the Pareto front. Moreover, these partial descent steps will only allow to explore, outwards, the extreme parts of the current front approximation, whereas the other descent step will mainly drive points to Pareto stationarity; as a result, even large holes within the current solutions set cannot be filled.

Taking the reasoning to the extreme, let us assume that the starting set of solutions already lies on the Pareto front; if the set contains only one point, then by repeated partial descent w.r.t. $\mathcal{I}_1$ and $\mathcal{I}_2$ the entire Pareto front can be spanned quite uniformly; this situation is depicted in Figure 6.1a. If, on the other hand, there are two starting solutions, possibly far away from each other in the objectives space, then only the extreme parts of the front will be spanned, while the gap between the two points is not tackled (Figure 6.1b). Of course, the same reasoning applies with more than two starting points.

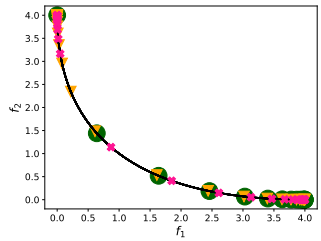
The paradoxical behavior of the algorithm is such that it might be convenient to start far away from the Pareto front. In this way, FSD may have many



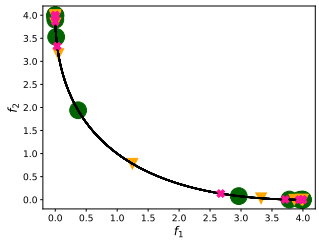
(a) FSD starts from 1 Pareto point.



(b) FSD starts from 2 Pareto points.



(c) 3 independent FSD runs, started from 3 different random points.



(d) 3 independent runs of FSD with the extrapolation strategy, started from the same 3 random points as in (c).

Figure 6.1: Pareto fronts obtained by the FSD algorithm on the convex JOS problem [51] ($n = 5$). For more details about the experimental settings, the reader is referred to Section 9.1.2.

iterations at its disposal to increase the size of the set X^k and uniformly span the objectives space; points are then driven to Pareto stationarity thanks to steps carried out considering $\mathcal{I} = \{1, 2\}$. Anyhow, the results are still influenced, somewhat randomly, by the starting solutions, as shown in Figure 6.1c. Moreover, the extreme parts of the front are always spanned much more densely than the central one. We shall remark that, as the intermediate regions of the front often provide the most interesting trade-offs to users, this is a very significant issue in practice.

The extrapolation technique proposed in [23] and mentioned in Remark 3.3 might allow to partly alleviate the issue discussed here, as much more nondominated solutions are obtained at each iteration; however, it is again

the exploration of the extreme regions that is mainly enhanced and sped up, with possibly overall counterproductive results (Figure 6.1d).

6.2 Improved Front Steepest Descent

In Algorithm 6.1, we report the scheme of a modified Front Steepest Descent (`IFSD`) algorithm that overcomes the limitations of Algorithm 3.4 discussed in Section 6.1.

Algorithm 6.1: Improved Front Steepest Descent (`IFSD`)

Input: $F : \mathbb{R}^n \to \mathbb{R}^m$, $X^0 \subset \mathbb{R}^n$ set of mutually nondominated points w.r.t. $F(\cdot)$, $\alpha_0 > 0$, $\delta \in (0,1)$.
$k = 0$
while *a stopping criterion is not satisfied* **do**
  $\hat{X}^k = X^k$
  forall $x_c \in X^k$ **do**
    if $x_c \in \hat{X}^k$ **then**
      if $\theta^{SD}(x_c) < 0$ **then**
        $\alpha_c^k = \texttt{ALS}(F(\cdot), \mathbb{R}^n, x_c, v^{SD}(x_c))$
      else
        $\alpha_c^k = 0$
      $z_c^k = x_c + \alpha_c^k v^{SD}(x_c)$
      $\hat{X}^k = \left(\hat{X}^k \cup \{z_c^k\}\right) \setminus \left\{y \in \hat{X}^k \mid F(z_c^k) \lneqq F(y)\right\}$
      forall $\mathcal{I} \subseteq \{1, \ldots, m\}$ *s.t.* $\theta_{\mathcal{I}}^{SD}(z_c^k) < 0$ **do**
        if $z_c^k \in \hat{X}^k$ **then**
          $\alpha_c^{\mathcal{I}} = \max_{h \in \mathbb{N}} \{\alpha_0 \delta^h \mid \forall\, y \in \hat{X}^k \; \exists j \in \{1, \ldots, m\} \text{ s.t. } f_j(z_c^k + \alpha_0 \delta^h v_{\mathcal{I}}^{SD}(z_c^k)) < f_j(y)\}$
          $\hat{X}^k = \left(\hat{X}^k \cup \left\{z_c^k + \alpha_c^{\mathcal{I}} v_{\mathcal{I}}^{SD}(z_c^k)\right\}\right) \setminus \left\{y \in \hat{X}^k \mid F\left(z_c^k + \alpha_c^{\mathcal{I}} v_{\mathcal{I}}^{SD}(z_c^k)\right) \lneqq F(y)\right\}$
  $X^{k+1} = \hat{X}^k$
  $k = k + 1$
return X^k

Algorithm 6.1 includes several modifications w.r.t. the original FSD approach:

- for any point in X^k that is still nondominated when it is considered for exploration, a preliminary steepest descent step is carried out; this step exploits a classical single point Armijo line search (Algorithm 3.2);
- further searches w.r.t. subsets of objectives start at the obtained point, as long as it is not dominated;
- for partial descent searches, we require the new point to be nondominated by all other points in $\hat{X}^k$.

The idea is that, with these modifications, all points may be used to start exploration based on partial descent; convergence of all the produced points towards stationarity is then forced by means of the "preliminary" steepest descent step, that ensures the sufficient decrease. In Figure 6.2, the behavior of the proposed approach in the same settings used for Figure 6.1 is shown. In this example we can observe that now, regardless of the starting point(s), the entire Pareto front is effectively spanned, with not even tiny holes.

In the next section we prove that the algorithm is well defined and actually produces convergent sequences of points.

6.2.1 Convergence Analysis

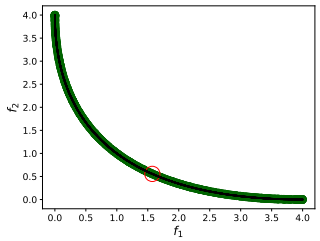
In this section, we provide the formal convergence analysis for Algorithm 6.1.

Proposition 6.1. *The line search at Step 8 of Algorithm 6.1 is well defined.*

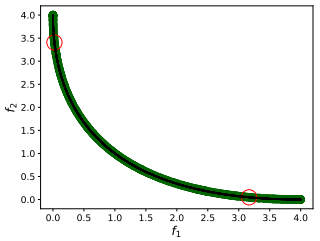
Proof. The result follows from Proposition 3.2 ($\Omega = \mathbb{R}^n$) and by the if condition at Step 7 that ensures that $\theta^{SD}(x_c) < 0$. □

Proposition 6.2. *Step 15 of Algorithm 6.1 is well defined if z_c^k is nondominated with respect to points in $\hat{X}^k$.*

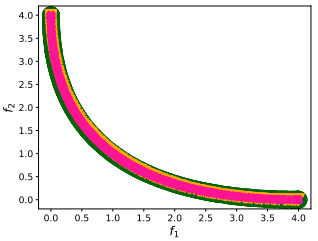
Proof. Let y be any point in $\hat{X}^k$; if $F(y) = F(z_c^k)$, then by [36, Lemma 4] and the condition $\theta_{\mathcal{I}}^{SD}(z_c^k) < 0$, there exists $\bar{\alpha} > 0$ such that $F_{\mathcal{I}}\left(z_c^k + \alpha v_{\mathcal{I}}^{SD}(z_c^k)\right) < F_{\mathcal{I}}(z_c^k) = F_{\mathcal{I}}(y)$ for all $\alpha < \bar{\alpha}$; thus there exists h sufficiently large such that $f_j\left(z_c^k + \alpha_0 \delta^h v_{\mathcal{I}}^{SD}(z_c^k)\right) < f_j(y)$ for all $j \in \mathcal{I}$. If, on the other hand, there exists $j \in \{1, \ldots, m\}$ such that $f_j(z_c^k) < f_j(y)$, then by the continuity of



(a) 1 Pareto point as in Figure 6.1a



(b) 2 Pareto points as in Figure 6.1b



(c) 3 independent runs from the same random points as those of Figure 6.1(c)-(d)

Figure 6.2: Pareto fronts obtained by the `IFSD` algorithm on the convex JOS problem [51] ($n = 5$). For more details about the experimental settings, the reader is referred to Section 9.1.2.

$F(\cdot)$ there exists $\alpha = \alpha_0 \delta^h$ sufficiently small such that $f_j\left(z_c^k + \alpha v_{\mathcal{I}}^{SD}(z_c^k)\right) < f_j(y)$. Thus, the condition can be satisfied for all $y \in \hat{X}^k$ and $\alpha_c^{\mathcal{I}}$ is the minimum of the corresponding values of $\alpha_0 \delta^h$. □

Proposition 6.3. *If X^k contains mutually nondominated points with respect to $F(\cdot)$, then $\hat{X}^k$ contains nondominated points at any time during iteration k; thus Step 15 of Algorithm 6.1 is always well defined and X^{k+1} is finally a set of nondominated solutions.*

Proof. See Appendix C. □

Lemma 6.1. *After Step 12 of Algorithm 6.1, z_c^k belongs to $\hat{X}^k$. Moreover, for all $\tilde{k} > k$, there exists $y \in X^{\tilde{k}}$ such that $F(y) \leq F(z_c^k)$.*

Proof. See Appendix C. □

In order to prove the convergence property of `IFSD`, we make use of Assumption 3.2 and the concept of linked sequence (Definition 3.1); both are also employed in [23] to prove the same property for `FSD` (Lemma 3.5).

Proposition 6.4. *Let Assumption 3.2 hold with $\Omega = \mathbb{R}^n$, the set X^0 and a point $x_0 \in X^0$. Let $\{X^k\}$ be the sequence of sets of nondominated points produced by Algorithm 6.1. Let $\{x_{j_k}\}$ be a linked sequence, then it admits accumulation points and every accumulation point is Pareto-stationary for Problem* (6.1).

Proof. For any k, either $x_0 \in X^k$ or $x_0 \notin X^k$. In the former case, since all points in X^k are mutually nondominated, we certainly have $x_{j_k} \in \overline{\mathcal{L}}_F(x_0)$. Otherwise, by a similar reasoning as in the proof of Lemma 6.1, we have that there is a point $y_k \in X^k$ such that $F(y_k) \leq F(x_0)$; since y_k does not dominate x_{j_k}, we have that there exists $h \in \{1, \ldots, m\}$ such that $f_h(x_{j_k}) \leq f_h(y_k) \leq f_h(x_0)$; thus, again, $x_{j_k} \in \overline{\mathcal{L}}_F(x_0)$. Therefore the entire sequence $\{x_{j_k}\}$ belongs to the compact set $\overline{\mathcal{L}}_F(x_0)$, and thus admits accumulation points.

Now, let us consider an accumulation point $\bar{x}$ of a linked sequence $\{x_{j_k}\}$, i.e., there exists $K \subseteq \{1, 2, \ldots\}$ such that

$$\lim_{\substack{k \to \infty \\ k \in K}} x_{j_k} = \bar{x}.$$

We assume by contradiction that $\bar{x}$ is not Pareto-stationary, i.e., by Lemma 2.2 with $\Omega = \mathbb{R}^n$ and $M_j(\cdot) = I_n$ for all $j \in \{1, \ldots, m\}$, $\theta^{SD}(\bar{x}) < 0$. Thus, by the continuity of $\theta^{SD}(\cdot)$ (Lemma 2.4), there exists $\varepsilon > 0$ such that for all $k \in K$ sufficiently large we have $\theta^{SD}(x_{j_k}) \leq -\varepsilon < 0$. Let $z_{j_k} = x_{j_k} + \alpha_{j_k} v^{SD}(x_{j_k})$ the point obtained at Step 11 of the algorithm starting from x_{j_k}. Now, $\alpha_{j_k} \in [0, \alpha_0]$, which is a compact set, thus there exists a further subsequence $K_1 \subseteq K$ such that $\alpha_{j_k} \to \bar{\alpha} \in [0, \alpha_0]$. Moreover, function $v^{SD}(\cdot)$ is continuous by Lemma 2.4, thus $v^{SD}(x_{j_k}) \to v^{SD}(\bar{x})$ for $k \to \infty$, $k \in K_1$. Hence, taking the limits along K_1 we also get that $z_{j_k} \to \bar{x} + \bar{\alpha} v^{SD}(\bar{x}) = \bar{z}$.

By the definition of α_{j_k}, z_{j_k} (Steps 8-11) and Remark 3.1 ($\Omega = \mathbb{R}^n$, $M_j(\cdot) = I_n$ for all $j \in \{1, \ldots, m\}$), we have that $F(z_{j_k}) < F(x_{j_k}) + \mathbf{1}_m \gamma \alpha_{j_k} \theta^{SD}(x_{j_k})$. Taking the limits for $k \in K_1$, $k \to \infty$, recalling again

the continuity of $\theta^{SD}(\cdot)$, we get

$$F(\bar{z}) \leq F(\bar{x}) + \mathbf{1}_m \gamma \bar{\alpha} \theta^{SD}(\bar{x}) \leq F(\bar{x}) - \mathbf{1}_m \gamma \bar{\alpha} \varepsilon. \tag{6.2}$$

Now, given $k \in K_1$, let $k_1(k)$ be the smallest index in K_1 such that $k_1(k) > k$. By Lemma 6.1, there exists $y_{j_{k_1(k)}} \in X^{k_1(k)}$ such that $F(y_{j_{k_1(k)}}) \leq F(z_{j_k})$; moreover, $x_{j_{k_1(k)}} \in X^{k_1(k)}$; by Proposition 6.3, the points in $X^{k_1(k)}$ are mutually nondominated, hence there exists $h(k) \in \{1, \ldots, m\}$ such that $f_{h(k)}(x_{j_{k_1(k)}}) \leq f_{h(k)}(y_{j_{k_1(k)}}) \leq f_{h(k)}(z_{j_k})$. Considering a further subsequence $K_2 \subseteq K_1$ such that $h(k) = h$ for all $k \in K_2$ and taking the limits, we obtain $f_h(\bar{x}) \leq f_h(\bar{z})$. Putting this last result together with (6.2), we get

$$f_h(\bar{x}) \leq f_h(\bar{z}) \leq f_h(\bar{x}) - \gamma \bar{\alpha} \varepsilon.$$

Since $\bar{\alpha} \in [0, \alpha_0]$, $\varepsilon > 0$ and $\gamma > 0$, the above chain of inequalities can only hold if $\bar{\alpha} = \lim_{k \to \infty, k \in K_2} \alpha_{j_k} = 0$. For all $k \in K_2$ sufficiently large, we have $\theta^{SD}(x_{j_k}) < 0$ and, thus, α_{j_k} is defined at Step 8. Since $\alpha_{j_k} \to 0$, for any $q \in \mathbb{N}$, for all $k \in K_2$ large enough we certainly have $\alpha_{j_k} < \alpha_0 \delta^q$; thus, the Armijo condition $F\left(x_{j_k} + \alpha v^{SD}(x_{j_k})\right) \leq F(x_{j_k}) + \gamma \alpha J_F(x_{j_k}) v^{SD}(x_{j_k})$ is not satisfied by $\alpha = \alpha_0 \delta^q$, i.e., there exists $\tilde{h}(k)$ such that

$$f_{\tilde{h}(k)}\left(x_{j_k} + \alpha_0 \delta^q v^{SD}(x_{j_k})\right) > f_{\tilde{h}(k)}(x_{j_k}) + \gamma \alpha_0 \delta^q \nabla f_{\tilde{h}(k)}(x_{j_k})^\top v^{SD}(x_{j_k}).$$

Taking the limits along a suitable subsequence such that $\tilde{h}(k) = \tilde{h}$, recalling the continuity of $F(\cdot)$, $J_F(\cdot)$ and $v^{SD}(\cdot)$, we get

$$f_{\tilde{h}}\left(\bar{x} + \alpha_0 \delta^q v^{SD}(\bar{x})\right) \geq f_{\tilde{h}}(\bar{x}) + \gamma \alpha_0 \delta^q \nabla f_{\tilde{h}}(\bar{x})^\top v^{SD}(\bar{x}).$$

Now, since q is arbitrary and $\theta^{SD}(\bar{x}) < 0$, i.e., $J_F(\bar{x}) v^{SD}(\bar{x}) < \mathbf{0}_m$, this is absurd by Proposition 3.2 ($\Omega = \mathbb{R}^n$). The proof is thus complete. □

6.3 Conclusions

In this chapter, we introduced an improved Front Steepest Descent algorithm with asymptotic convergence guarantees similar to those of the original method. The novel algorithm is designed so as to overcome some empirically evident limitation of `FSD`, that is often unable to span large portions of the Pareto front. Numerical evidence (Section 9.4) suggests that the proposed procedure effectively achieves this goal.

Future work should be focused on the integration of the proposed approach and the extrapolation strategy proposed in [23].

Chapter 7

Pareto Front Approximation through a Multi-objective Augmented Lagrangian Method

In this chapter [1], we consider smooth multi-objective optimization problems with convex constraints, i.e.,

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \; & F(x) = (f_1(x), \ldots, f_m(x))^\top \\ \text{s.t.} \; & g(x) \leq \mathbf{0}_p, \end{aligned} \tag{7.1}$$

where $g : \mathbb{R}^n \to \mathbb{R}^p$ is a continuously differentiable, component-wise convex function. In line with Problem (2.1), we denote the feasible set as $\Omega = \{x \in \mathbb{R}^n \mid g(x) \leq \mathbf{0}_p\}$. In the following, we also indicate with $J_g(\cdot) = (\nabla g_1(\cdot), \ldots, \nabla g_p(\cdot))^\top \in \mathbb{R}^{p \times n}$ the Jacobian matrix associated with $g(\cdot)$. Note that equality constraints can be equivalently expressed as couples of opposite inequality constraints, so this formulation is in fact general. Actually, specific management of equality constraints can often be convenient from a computational perspective; the following discussion could easily be extended to address the presence of explicit equality constraints, but we prefer not to take them into account for the sake of simplicity.

The contribution of this chapter consists of the definition of an extended version of the augmented Lagrangian algorithm for multi-objective optimiza-

[1] Part of the content of this chapter has been published as "Pareto front approximation through a multi-objective augmented Lagrangian method" in *EURO Journal on Computational Optimization, 2021* [22].

tion (ALAMO) proposed in [21], which deals with sets of points and effectively produces an approximation of the Pareto front for constrained vector-valued problems. The key elements that characterize the proposed algorithm are:

- the management of a set of points at each iteration which are all mutually nondominated w.r.t. the current augmented Lagrangian;
- the use of partial descent directions w.r.t. subsets of objectives in order to enrich the approximate front;
- the use of a common penalty parameter and Lagrange multipliers for all points in the set of solutions;
- the use of the MOSD algorithm (Section 3.1.1) to make each point in the current set approximately Pareto-stationary w.r.t. the augmented Lagrangian, with increasing accuracy throughout the iterations.

For the proposed algorithm, we prove properties of convergence to Pareto-stationarity for the generated sequence of sets of points, without the need to recur to the concept of linked sequence introduced in [67]. In fact, the convergence along linked sequences is implied by our result.

To the best of our knowledge, the SQP procedure [37] is the only other derivative based method for constructing an approximated Pareto front of constrained multi-objective problems that can be found in the literature. It is worth remarking that, in contrast with the SQP method, convergence of our algorithm does not depend on a final refinement step that follows a finite exploration phase. As also noted by its authors, SQP can indeed be seen as a single point procedure run in a multi-start fashion. On the contrary, in our procedure convergence and exploration advance alongside, with both asymptotically improving.

In order to introduce our new approach, we recall the definition of multi-objective augmented Lagrangian for problems with inequality constraints.

Definition 7.1 ([21])**.** The *multi-objective augmented Lagrangian* function with penalty parameter $\tau > 0$ associated with Problem (7.1) is given by

$$\mathcal{L}^{\tau}(x, \mu) = F(x) + \mathbf{1}_m \frac{\tau}{2} \left(\sum_{i=1}^{p} \left(\max \left\{ 0, g_i(x) + \frac{\mu_i}{\tau} \right\} \right)^2 \right), \tag{7.2}$$

where $x \in \mathbb{R}^n$ and $\mu \geq \mathbf{0}_p$ is the vector of Lagrange multipliers.

7.1 The Algorithm

In this section, we describe the front-oriented multi-objective augmented Lagrangian method, which we call `FRONT-ALAMO`, to solve Problem (7.1). The algorithmic scheme is reported in Algorithm 7.1. Note that we have denoted by $\texttt{MOSD}(\cdot,\cdot,\cdot,\varepsilon_k)$ the `MOSD` algorithm (Section 3.1.1) run until the solution is ε_k-Pareto-stationary. We also denote by $v_k^{SD}(\cdot)$, $\theta_k^{SD}(\cdot)$ and $v_{k,\mathcal{I}}^{SD}(\cdot)$, $\theta_{k,\mathcal{I}}^{SD}(\cdot)$, with $\mathcal{I} \subseteq \{1,\dots,m\}$, the functions of the steepest common and partial descent directions (see Table 2.1) associated with $\mathcal{L}^{\tau_k}(\cdot,\mu^k)$, respectively.

Through the iterations, the algorithm produces a sequence of sets of points $\{X^k\}$, which approximate the Pareto set of the original problem with increasing accuracy. At each iteration, an augmented Lagrangian function defined as in (7.2) is considered, with penalty parameter τ_k and multipliers μ^k. At the beginning of the generic iteration k, all points that are dominated w.r.t. $\mathcal{L}^{\tau_k}(\cdot,\mu^k)$ are filtered out of the set; we denote such filtered set by $\hat{X}^k$. Now, the following iterate X_{tmp} is initialized as $\hat{X}^k$. Then, each point $x_c \in \hat{X}^k$ is used as a starting point for exploration as long as it is non-dominated: in particular, for any possible subset $\mathcal{I} \subseteq \{1,\dots,m\}$ the steepest partial descent direction, provided that it actually exists, is exploited to obtain a new point that must be not dominated w.r.t. $\mathcal{L}^{\tau_k}(\cdot,\mu^k)$ by any other point in X_{tmp} (Line 9). This point is then refined by means of the `MOSD` procedure up to ε_k-Pareto-stationarity (Line 10) and, finally, it is added in X_{tmp} and all the solutions dominated by it are removed (Line 11). After considering all the possible subsets $\mathcal{I} \subseteq \{1,\dots,m\}$, an additional search based on the steepest common descent direction is started from the solution x_c: if such a direction exists, a classical Armijo-type line search (Algorithm 3.2) is performed (Line 13); the new point is then refined through the `MOSD` procedure (Line 14); if the resulting solution is not dominated w.r.t. $\mathcal{L}^{\tau_k}(\cdot,\mu^k)$ by any other point in X_{tmp}, it is consequently added to such set, while all the points that are dominated by it are removed (Line 16).

Once all points in $\hat{X}^k$ are tested, the constructed set will constitute the next iterate X^{k+1}. The multipliers and the penalty parameter are updated similarly as in the scalar `ALM` with multipliers safeguarding [54], with one key adjustment: to evaluate how much a constraint is violated, the worst violation attained on that constraint by any point in X^{k+1} is considered. In addition, the second clause of the conditional statement at Line 21 allows to avoid unfortunate cases where a point which is strictly feasible w.r.t. some constraint $g_i(\cdot)$ is unnecessarily pushed to satisfy it with a larger margin.

Algorithm 7.1: Front-oriented Multi-Objective Augmented Lagrangian (`FRONT-ALAMO`)

Input: $\mu^0 \in \mathbb{R}^p_+$, $\bar{\mu} \geq 0$, $\rho > 1$, $\sigma \in (0,1)$, $\tau_0 > 0$, Ω feasible set, $X^0 \subset \Omega$, $\{\varepsilon_k\} \subset \mathbb{R}^+_0$ a decreasing sequence, $\alpha_0 > 0$, $\delta \in (0,1)$.
for $k = 0, 1, \ldots$ **do**
  Let $\mathcal{L}^{\tau_k}(\cdot, \mu^k)$ be the current Augmented Lagrangian function defined as in (7.2)
  set $\hat{X}^k = X^k \setminus \{x \in X^k \mid \exists\, y \in X^k \text{ s.t. } \mathcal{L}^{\tau_k}(y, \mu^k) \lneqq \mathcal{L}^{\tau_k}(x, \mu^k)\}$
  set $X_{tmp} = \hat{X}^k$
  for $x_c \in \hat{X}^k$ **do**
    for $\mathcal{I} \in 2^{\{1,\ldots,m\}}$ **do**
      if $x_c \in X_{tmp} \wedge \theta^{SD}_{k,\mathcal{I}}(x_c) < 0$ **then**
        set $\alpha^{\mathcal{I}}_c = \max_{h \in \mathbb{N}}\{\alpha_0 \delta^h \mid \forall y \in X_{tmp}\ \exists j \in \{1,\ldots,m\} \text{ s.t. } \mathcal{L}^{\tau_k}_j(x_c + \alpha_0 \delta^h v^{SD}_{k,\mathcal{I}}(x_c), \mu^k) < \mathcal{L}^{\tau_k}_j(y, \mu^k)\}$
        set $z^{\mathcal{I}}_c = \texttt{MOSD}(\mathcal{L}^{\tau_k}(\cdot, \mu^k), \mathbb{R}^n, x_c + \alpha^{\mathcal{I}}_c v^{SD}_{k,\mathcal{I}}(x_c), \varepsilon_k)$
        set $X_{tmp} = \left(X_{tmp} \cup \{z^{\mathcal{I}}_c\}\right) \setminus \{y \in X_{tmp} \mid \mathcal{L}^{\tau_k}(z^{\mathcal{I}}_c, \mu^k) \lneqq \mathcal{L}^{\tau_k}(y, \mu^k)\}$
    if $\theta^{SD}_k(x_c) < 0$ **then**
      set $\alpha_c = \texttt{ALS}(\mathcal{L}^{\tau_k}(\cdot, \mu^k), \mathbb{R}^n, x_c, v^{SD}_k(x_c))$
      set $z_c = \texttt{MOSD}(\mathcal{L}^{\tau_k}(\cdot, \mu^k), \mathbb{R}^n, x_c + \alpha_c v^{SD}_k(x_c), \varepsilon_k)$
      if $\nexists\, y \in X_{tmp} : \mathcal{L}^{\tau_k}(y, \mu^k) \lneqq \mathcal{L}^{\tau_k}(z_c, \mu^k)$ **then**
        set $X_{tmp} = (X_{tmp} \cup \{z_c\}) \setminus \{y \in X_{tmp} \mid \mathcal{L}^{\tau_k}(z_c, \mu^k) \lneqq \mathcal{L}^{\tau_k}(y, \mu^k)\}$
  set $X^{k+1} = X_{tmp}$
  for $i = 1, \ldots, p$ **do**
    set $V^{k+1}_i = \min\left\{\min_{x \in X^{k+1}}\{-g_i(x)\}, \frac{\mu^k_i}{\tau_k}\right\}$
    set $\mu^{k+1}_i = \max\left\{0, \min\{\mu^k_i + \tau_k \max_{x \in X^{k+1}}\{g_i(x)\}, \bar{\mu}\}\right\}$
  if $||V^{k+1}|| > \sigma||V^k|| \vee (\exists\, x_{k+1} \in X^{k+1} \text{ s.t. } g_i(x_{k+1}) < 0 \wedge \mu^k_i + \tau_k g_i(x_{k+1}) > 0 \text{ for some } i \in \{1,\ldots,p\})$ **then**
    set $\tau_{k+1} = \rho\tau_k$
  else
    set $\tau_{k+1} = \tau_k$
return X^k

Remark 7.1. At each iteration k, the set X^{k+1} is a list of mutually nondominated points w.r.t. $\mathcal{L}^{\tau_k}(\cdot, \mu^k)$. As we will shortly see, maintaining a set of mutually nondominated points with respect to the augmented Lagrangian does not provide theoretical asymptotic properties. However, this has a remarkable impact from a computational point of view: it allows, especially at late iterations, to remove solutions that are too far from feasibility or that have bad values for all the objectives; in addition, in practice the algorithm will be run for a large enough number of iterations and then stopped; the solutions in the returned set are mutually nondominated w.r.t. the final augmented Lagrangian; because of this property, most of the points in the returned set that are "sufficiently feasible" are nondominated also w.r.t. the original problem.

In the next section, we will show in detail that Algorithm 7.1 is well defined and we will carefully address its convergence properties.

7.2 Convergence Analysis

In this section, we provide a rigorous formal analysis of Algorithm 7.1 from a theoretical perspective. We first show that the procedure is actually well defined and then we state its asymptotic convergence properties. In the analysis, we make use of Assumption 3.1 with $\Omega = \mathbb{R}^n$.

Concerning algorithm well-definiteness, we begin by noting that, having Assumption 3.1, by Lemma 3.1 and Proposition 3.1 ($\Omega = \mathbb{R}^n$, $M_j(\cdot) = I_n$ for all $j \in \{1, \ldots, m\}$) the `MOSD` procedure employed in Lines 10-14 is well-defined. The line search procedures at Lines 9-13 stop in a finite time too, producing a valid step size. The well-definiteness of the first one comes by Proposition 6.2 and the first condition in Line 8 that assures x_c is nondominated w.r.t. $\mathcal{L}^{\tau_k}(\cdot, \mu^k)$ in X_{tmp}. As for the second line search, the result holds straightforwardly from Proposition 3.2 ($\Omega = \mathbb{R}^n$), also recalling that the procedure starts at a point x_c such that $\theta_k^{SD}(x_c) < 0$, i.e., $J_F(x_c)v_k^{SD}(x_c) < \mathbf{0}_m$. Thus, we conclude that Algorithm 7.1 is well-defined.

Now, we are able to characterize the points belonging to each iterate set X^k.

Proposition 7.1. *Let $\{X^{k+1}\}$ be the sequence of sets generated by Algorithm 7.1. Then, for each k and for each $x_{k+1} \in X^{k+1}$, we have:*

1. *x_{k+1} is not dominated by any other point in X^{k+1} w.r.t. $\mathcal{L}^{\tau_k}(\cdot, \mu^k)$, i.e., there does not exist $y \in X^{k+1}$ such that $\mathcal{L}^{\tau_k}(y, \mu^k) \lneqq \mathcal{L}^{\tau_k}(x_{k+1}, \mu^k)$;*

2. *x_{k+1} is ε_k-Pareto-stationary w.r.t. $\mathcal{L}^{\tau_k}(\cdot, \mu^k)$.*

Proof. See Appendix C. □

Let $\{X^k\}$ be the sequence of (finite) sets produced by the algorithm. In order to assess the asymptotic convergence properties of Algorithm 7.1, we need to consider sequences of points $\{x_k\}$ such that $x_k \in X^k$ for all k.

We are now able to begin the convergence analysis with a technical lemma.

Lemma 7.1. *Let $\{X^k\}$ be the sequence of sets generated by Algorithm 7.1, and let $\{x_k\}$ be any sequence of points such that $x_k \in X^k$ for all k. Let $\bar{x}$ be a limit point of $\{x_k\}$, i.e., there exists an infinite subset $K \subseteq \{0, 1, \ldots\}$ such that*

$$\lim_{\substack{k \to \infty \\ k \in K}} x_k = \bar{x},$$

and suppose that $g(\bar{x}) \leq \mathbf{0}_p$, i.e., $\bar{x} \in \Omega$. Then, for all $i = 1, \ldots, p$ such that $g_i(\bar{x}) < 0$ we have

$$\max\{0, \mu_i^k + \tau_k g_i(x_{k+1})\} = 0$$

for all $k \in K$ sufficiently large.

Proof. See Appendix C. □

Next, we prove feasibility of limit points of all possible points sequences $\{x_k\}$ produced by the algorithm.

Proposition 7.2. *Let $\{X^k\}$ be the sequence of sets generated by Algorithm 7.1, with $\varepsilon_k \to 0$, and let $\{x_k\}$ be any sequence of points such that $x_k \in X^k$ for all k. Let $\bar{x}$ be a limit point of $\{x_k\}$. Then, $\bar{x}$ is feasible for Problem (7.1), i.e., $g(\bar{x}) \leq \mathbf{0}_p$.*

Proof. Let $K \subseteq \{0, 1, \ldots\}$ be an infinite subset such that

$$\lim_{\substack{k \to \infty \\ k \in K}} x_{k+1} = \bar{x}.$$

If the sequence $\{\tau_k\}$ is bounded, from the instructions of the algorithm there must exist k_1 such that, for all $k > k_1$, we have $\|V^{k+1}\| \leq \sigma \|V^k\|$. Since

$\sigma < 1$, this implies $\lim_{k\to\infty} \|V^k\| = 0$, i.e., for all $i \in \{1, \ldots, p\}$,

$$\lim_{k\to\infty} V_i^{k+1} = \lim_{k\to\infty} \min\left\{ \min_{x\in X^{k+1}} \{-g_i(x)\}, \frac{\mu_i^k}{\tau_k} \right\} = 0.$$

Since by definition $\mu_i^k \geq 0$ for all i and k, it has to be $\lim_{k\to\infty} \min_{x\in X^{k+1}} \{-g_i(x)\} \geq 0$. But $\min_{x\in X^{k+1}} \{-g_i(x)\} \leq -g_i(x_{k+1})$. Hence

$$g_i(\bar{x}) = \lim_{\substack{k\to\infty \\ k\in K}} g_i(x_{k+1}) \leq \lim_{\substack{k\to\infty \\ k\in K}} \max_{x\in X^{k+1}} \{-g_i(x)\} \leq 0.$$

Now, assume $\tau_k \to \infty$. Let us suppose, by contradiction, that there exists $\hat{d} \in \mathbb{R}^n$ such that

$$\max_{j=1,\ldots,m} \left\{ \left(\sum_{i=1}^{p} \max\{0, g_i(\bar{x})\} \nabla g_i(\bar{x}) \right)^\top \hat{d} \right\} < 0. \tag{7.3}$$

From Proposition 7.1, we know that each point $x \in X^{k+1}$ is ε_k-Pareto-stationary w.r.t. $\mathcal{L}^{\tau_k}(\cdot, \mu^k)$. Hence, $\forall\, d \in \mathbb{R}^n$, $\max_{j=1,\ldots,m} \nabla \mathcal{L}_j^{\tau_k}(x_{k+1}, \mu^k)^\top d + \frac{1}{2}\|d\|^2 \geq -\varepsilon_k$. Considering the last equation and the direction $\hat{d} \in \mathbb{R}^n$, we have that

$$\max_{j=1,\ldots,m} \left\{ \left(\nabla f_j(x_{k+1}) + \tau_k \sum_{i=1}^{p} \max\left\{0, g_i(x_{k+1}) + \frac{\mu_i^k}{\tau_k}\right\} \nabla g_i(x_{k+1}) \right)^\top \hat{d} \right\} + \frac{1}{2}\|\hat{d}\|^2 \geq -\varepsilon_k.$$

Dividing both sides of the inequality by τ_k and taking the limits for $k \to \infty$, $k \in K$, recalling the continuity of $J_F(\cdot)$ and $J_g(\cdot)$, the boundedness of $\{\mu^k\}$ and that $\tau_k \to \infty$, we get

$$\max_{j=1,\ldots,m} \left\{ \left(\sum_{i=1}^{p} \max\{0, g_i(\bar{x})\} \nabla g_i(\bar{x}) \right)^\top \hat{d} \right\} \geq 0,$$

which is in contradiction with (7.3). Thus, we conclude that, $\forall d \in \mathbb{R}^n$,

$$\max_{j=1,\ldots,m} \left\{ \left(\sum_{i=1}^{p} \max\{0, g_i(\bar{x})\} \nabla g_i(\bar{x}) \right)^\top d \right\} \geq 0,$$

which, since the arguments of the outer max operator are independent of j, is equal to $\frac{1}{2}\nabla \left(\|\max\{\mathbf{0}_p, g(\bar{x})\}\|^2 \right)^\top d \geq 0,\ \forall\, d \in \mathbb{R}^n$, where the max operator

is intended component-wise. Thus, $\bar{x}$ is a critical point for problem

$$\min_{x \in \mathbb{R}^n} \frac{1}{2} \| \max\{\mathbf{0}_p, g(x)\} \|^2.$$

Since $\Omega \neq \emptyset$ and the above problem is convex, $\bar{x}$ is a global minimum point with $\max\{\mathbf{0}_p, g(\bar{x})\} = \mathbf{0}_p$, i.e., $g(\bar{x}) \leq \mathbf{0}_p$. □

Finally, we show that limit points are Pareto-stationary for the original problem.

Proposition 7.3. *Let $\{X^k\}$ be the sequence of sets generated by Algorithm 7.1, with $\varepsilon_k \to 0$, and let $\{x_k\}$ be any sequence of points such that $x_k \in X^k$ for all k. Let $\bar{x}$ be a limit point of $\{x_k\}$. Then, $\bar{x}$ is Pareto-stationary for Problem* (7.1).

Proof. Recalling that, from Proposition 7.1, x_{k+1} is ε_k-Pareto-stationary for $\mathcal{L}^{\tau_k}(\cdot, \mu^k)$, the result similarly follows as in Proposition 6 from [21], where Lemma 7.1 can be used in place of Lemma 9 from the referenced paper. □

Remark 7.2. Pareto-stationarity, which we are able to prove for limit points of `FRONT-ALAMO`, is the same property that holds for limit points of the sequence produced by the single point `ALAMO` and analogous, in the scalar context, to stationarity attained by limit points of scalar `ALM`. Therefore, it is reasonable to assume that stronger properties are unlikely to be obtained by an `ALM`-like algorithm.

Remark 7.3. In the literature of Pareto front constructing descent methods [23, 67], convergence analysis is based on the concept of linked sequence (Definition 3.1). It is easy to see that linked sequences are a particular instance of the sequences of points considered in Propositions 7.2-7.3, hence the convergence result obtained for Algorithm 7.1 is somewhat stronger than those based on linked sequences.

Remark 7.4. In our theoretical analysis we assumed the existence of a limit point $\bar{x}$. As commonly done in the literature of augmented Lagrangian methods [12, 21], we do not directly address properties of existence of limit points, leaving it to boundedness arguments on the sequences, level sets, lower-level feasible sets or restart strategies.

Remark 7.5. The `SQP` algorithm [37], which is, to the best of our knowledge, the only other derivative-based method in the literature to generate

an approximation of the Pareto front of MOO problems with general convex constraints, has similar convergence properties as Algorithm 7.1, in the sense that limit points of sequences of solutions are Pareto-stationary. However, the setting is basically different, as the exploration phase of the SQP method is eventually stopped and all the obtained points are then independently driven to Pareto-stationarity by an iterative method. Convergence hence follows from a single-point mechanism. On the other hand, in Algorithm 7.1 exploration and convergence are performed somewhat in parallel, in an effectively multiple-points fashion.

7.3 Conclusions

In this chapter, we considered smooth multi-objective optimization problems subject to convex constraints. We focused on the task of generating good Pareto front approximations for this class of problems and, after a brief review of the existing literature, we proposed an Augmented Lagrangian Method specifically designed for this task.

The method represents an extension of the ALAMO procedure [21], which is designed to produce a single Pareto-stationary solution. The proposed algorithm handles, at each iteration, a list of points that are mutually non-dominated and Pareto-stationary with respect to the current multi-objective augmented Lagrangian. Line searches along steepest common and partial descent directions are employed to carry out an exploration of the objectives space. The penalty parameter and the Lagrange multipliers are updated taking into account constraint violations by all the points in the current list. For this algorithm, we proved global convergence to Pareto-stationarity of the sequences of points in the iterates lists. This type of convergence is more general than that based on linked sequences.

Thorough computational experiments (Section 9.5) show that our method outperforms the SQP algorithm [37] in terms of popular metrics for multi-objective optimization. Moreover, we compared the proposed procedure with the state-of-the-art derivative-free (DMS [25]) and genetic (NSGA-II [26]) approaches. Our procedure proved to obtain better results even w.r.t. the two mentioned ones.

Chapter 8

Cardinality-Constrained Multi-Objective Optimization: Novel Optimality Conditions and Algorithms

In this chapter, we consider multi-objective optimization problems with a sparsity constraint on the vector of variables:

$$\begin{aligned} \min_{x \in \mathbb{R}^n} \; & F(x) = (f_1(x), \ldots, f_m(x))^\top \\ \text{s.t.} \; & \|x\|_0 \leq s, \end{aligned} \tag{8.1}$$

where $\|\cdot\|_0$ denotes the ℓ_0 pseudo-norm, i.e., the number of nonzero components of a vector, and $s \in \mathbb{N}$ is such that $1 \leq s < n$. In line with Problem (2.1), we denote the feasible set induced by the upper bound on the ℓ_0 pseudo-norm as $\Omega = \{x \in \mathbb{R}^n \mid \|x\|_0 \leq s\}$.

Problems where solutions with few nonzero components are required represent a topic recently investigated by the optimization community [99]. Solution sparsity is often induced by the direct introduction of a cardinality constraint on the variables vector. However, setting an upper-bound for the ℓ_0 pseudo-norm makes the problem partly combinatorial and thus $\mathcal{NP}$-hard [11, 82]. For this reason, many approaches whose aim is to solve this problem approximately have been proposed. We refer the reader to [99] for a thorough survey of these methods. However, algorithms dealing exactly

with the ℓ_0 pseudo-norm can be found in the literature. In particular, the *Iterative Hard Thresholding* (IHT) algorithm [3], the *Penalty Decomposition* (PD) approach [70] and the *Sparse Neighborhood Search* (SNS) method [58] are designed to be employable in the most general cases, even without convexity assumptions. With these methods, problems are tackled by means of continuous local optimization steps and convergence to solutions satisfying necessary optimality conditions is guaranteed.

Although MOO and problems requiring solution sparsity have been thoroughly investigated separately, the combination of these two themes, i.e., sparsity and multiple objectives, has almost never been explored. The theoretical foundation for cardinality-constrained MOO was recently laid in [57], where a Penalty Decomposition approach was also proposed for sparse MOO tasks along with its convergence analysis. Moreover, a theoretical study extending the work from [15] to the MOO case was presented in [41]. The development of high-performing procedures to deal with this class of problems is beneficial for many real-world applications. For instance, there are several reasons in machine learning for requiring sparsity within classification/regression models (e.g., interpretability [6], robustness [104], lightness [18]). In addition, there are approaches in the literature where learning tasks can be tackled from a Pareto-based, multi-objective perspective: fitting quality and reducing model complexity (e.g., minimizing $\|w\|^2$, with w being the model weights vector) are just two examples of conflicting objectives for which a good trade-off may be useful [52].

In this chapter, we continue the theoretical analysis started in [57], introducing new optimality conditions for MOO problems with cardinality constraints. In particular, we define the concept of L-stationarity in MOO, which is directly inspired by the homonymous condition for sparse single-objective optimization (SOO) tasks [3]. Then, we introduce two new algorithms to solve these problems; the first one consists of an extension of the IHT method to the MOO case and it is designed to retrieve an L-stationary solution; we call this method *Multi-Objective Iterative Hard Thresholding* (MOIHT) and we prove that it is indeed guaranteed to converge to points satisfying the newly introduced necessary optimality condition. The second algorithm, on the other hand, is a two-stage approach whose ultimate goal is to approximate the whole Pareto front. This method, which we call *Sparse Front Steepest Descent* (SFSD), is theoretically analyzed too.

Additional Notation Given an index set $S \subseteq \{1, \dots, n\}$, the cardinality of S is indicated with $|S|$, while we denote by $\bar{S} = \{1, \dots, n\} \setminus S$ its *complementary set*; we call S a *singleton* if $|S| = 1$. Letting $x \in \mathbb{R}^n$, we denote by x_S the sub-vector of x induced by S, i.e., the vector composed by the components x_i, with $i \in S$; $S_1(x) = \{i \in \{1, \dots, n\} \mid x_i \neq 0\}$ represents the *support set* of x, that is, the set of the indices corresponding to the non-zero components of x; $S_0(x) = \{1, \dots, n\} \setminus S_1(x)$ is the $S_1(x)$ complementary set. Furthermore, according to [4], we say that an index set J is a *super support set* for x if $S_1(x) \subseteq J$ and $|J| = s$; the set of all super support sets at x is denoted by $\mathcal{J}(x)$ and it is a singleton if and only if $\|x\|_0 = s$.

8.1 Preliminaries: the Proximal Operator in MOO

A thorough analysis of proximal methods in the multi-objective setting can be found in the literature (see, e.g., [13, 97]). For the scope of this work, we refer to the discussion carried out in [97], where the considered MOO problems are of the form

$$\min_{x \in \mathbb{R}^n} \left(f_1(x) + g_1(x), \dots, f_m(x) + g_m(x)\right)^\top. \tag{8.2}$$

For all $j \in \{1, \dots, m\}$, $f_j(\cdot)$ is assumed to be continuously differentiable, whereas $g_j(\cdot)$ is lower semi-continuous, proper convex but not necessarily smooth.

Let $x_k \in \mathbb{R}^n$. A *proximal step* at x_k can be carried out according to $x_{k+1} = x_k + t_k d_k$, where t_k is a suitable stepsize and the descent direction d_k is obtained solving

$$\min_{d \in \mathbb{R}^n} \max_{j \in \{1, \dots, m\}} \left\{\nabla f_j(x_k)^\top d + g_j(x_k + d) - g_j(x_k)\right\} + \frac{L}{2}\|d\|^2, \tag{8.3}$$

where $L > 0$. An optimal solution of Problem (8.2) is such that $\mathbf{0}_n$ is solution to (8.3).

Interestingly and similarly to the scalar case, Problem (8.3) can be seen as a generalization of well-known schemes to define the search direction:

- if, for all $j \in \{1, \dots, m\}$, $g_j(\cdot) = 0$, then (8.3) coincides with the problem of finding the steepest common descent direction [36];

- if, for all $j \in \{1, \dots, m\}$, $g_j(\cdot)$ is the indicator function of a convex set C, then (8.3) becomes equivalent to the constrained steepest common descent direction problem [29].

The mentioned search directions can be also found listed in Table 2.1 of this manuscript.

In the next section, we are going to show that the proximal operator can be used to handle the nonconvex set Ω, in line with the work [3] for scalar optimization.

8.2 Optimality Conditions

Under differentiability assumptions on the objective function $F(\cdot)$, a Pareto-stationarity condition was proved in [57] to be necessary for (local) weak Pareto optimality. In what follows, we report slightly different definition and properties, adapted to Problem (8.1).

Definition 8.1 ([57, Definition 3.2])**.** A point $\bar{x} \in \Omega$ is Pareto-stationary for (8.1) if

$$\theta^S(\bar{x}) = \min_{d \in \mathcal{D}(\bar{x})} \max_{j=1,\dots,m} \nabla f_j(\bar{x})^\top d + \frac{1}{2}\|d\|^2 = 0, \tag{8.4}$$

where $\mathcal{D}(\bar{x}) = \{v \in \mathbb{R}^n \mid \exists \bar{t} > 0 : \bar{x} + tv \in \Omega \ \forall t \in [0, \bar{t}\,]\} = \{v \in \mathbb{R}^n \mid \|v_{S_0(\bar{x})}\|_0 \leq s - \|\bar{x}\|_0\}$ is the set of feasible directions at $\bar{x}$.

We denote by $\mathcal{V}^S(\bar{x})$ the set of optimal solutions of Problem (8.4) at $\bar{x}$.

Lemma 8.1 ([57, Proposition 3.3])**.** *Let $\bar{x} \in \Omega$ be locally weakly Pareto optimal for Problem* (8.1)*. Then, $\bar{x}$ is Pareto-stationary for* (8.1)*.*

The second lemma states that, assuming the convexity of the objective functions, the stationarity condition is also sufficient for local weak Pareto optimality.

Lemma 8.2. *Assume $F(\cdot)$ is component-wise convex. Let $\bar{x} \in \Omega$ a Pareto-stationary point for Problem* (8.1)*. Then, $\bar{x}$ is locally weakly Pareto optimal for* (8.1)*.*

Proof. See Appendix C. □

Moreover, in [57], the Lu-Zhang first-order optimality conditions for scalar cardinality-constrained problems [70] have been extended to the multi-objective optimization setting.

Definition 8.2 ([57, Definition 3.6])**.** A point $\bar{x} \in \Omega$ satisfies the *Multi-Objective Lu-Zhang first-order optimality conditions* (MOLZ conditions) for (8.1) if there exists a super support set $J \in \mathcal{J}(\bar{x})$ such that

$$\theta^J(\bar{x}) = \min_{d \in \mathbb{R}^n} \max_{j=1,\dots,m} \nabla f_j(\bar{x})^\top d + \frac{1}{2}\|d\|^2 = 0 \quad \text{s.t.} \quad d_{\bar{J}} = \mathbf{0}_{|\bar{J}|} \tag{8.5}$$

Since Problem (8.5) has a strongly convex objective function and a convex feasible set, it has a unique optimal solution at $\bar{x}$ that we indicate with $v^J(\bar{x})$.

Lemma 8.3 ([57, Proposition 3.7])**.** *Let $\bar{x} \in \Omega$ be a Pareto stationary point for Problem* (8.1)*. Then, $\bar{x}$ satisfies the MOLZ conditions.*

As pointed out in [57], the converse is not always true; in order to obtain an equivalence between the two conditions, we need a stronger requirement.

Lemma 8.4 ([57, Proposition 3.10])**.** *A point $\bar{x} \in \Omega$ is a Pareto stationary point for Problem* (8.1) *if and only if it satisfies the MOLZ conditions for all $J \in \mathcal{J}(\bar{x})$.*

The Pareto-stationarity condition can be interpreted as a direct extension of the *basic feasibility* concept in cardinality-constrained SOO [3,4]. As such, the limitations of scalar basic feasibility naturally get transferred to the MOO case; in particular, Pareto-stationarity is only a local optimality condition and it does not allow to obtain information about the quality of the current support set. The MOLZ conditions emphasize this issue even more, being generally less restrictive than Pareto-stationarity.

With the above consideration in mind, we are motivated to extend the stronger L-stationarity condition from [3] to the MOO case. In order to do so, we shall reinterpret L-stationarity in terms of proximal operators. Specifically, we can employ Problem (8.3) to define L-stationarity for MOO.

Let us consider the problem

$$\theta^L(\bar{x}) = \min_{d \in \mathcal{D}^L(\bar{x})} \max_{j=1,\dots,m} \nabla f_j(\bar{x})^\top d + \frac{L}{2}\|d\|^2, \tag{8.6}$$

where $\mathcal{D}^L(\bar{x}) = \{v \in \mathbb{R}^n \mid \bar{x} + v \in \Omega\}$ and let us denote by $\mathcal{V}^L(\bar{x})$ the set of optimal solutions at $\bar{x}$ (since Ω is not a convex set, the solution is not

necessarily unique). It is easy to notice that Problem (8.6) is equivalent to (8.3) where, for all $j \in \{1, \dots, m\}$, $g_j(\cdot)$ is the indicator function of the set Ω.

Lemma 8.5. *Let $\bar{x} \in \Omega$ and $L > 0$. Then, the following conditions hold:*

1. *$\theta^L(\bar{x})$ and $\mathcal{V}^L(\bar{x})$ are well-defined;*
2. *$\theta^L(\bar{x}) \leq 0$;*
3. *the mapping $x \to \theta^L(x)$ is continuous.*

Proof. For Points 1. and 2. we can follow the proof of Points 1. and 2. of Lemma 2.2, having in this case $M_j(\cdot) = LI_n$ for all $j \in \{1, \dots, m\}$ and recalling that, by definition, $\mathcal{D}^L(\bar{x})$ is closed and non-empty.

The proof of Point 3. is identical to the one of Proposition 4 in [29]. The argument is not spoiled by the set $\mathcal{D}^L(\bar{x})$ being nonconvex. □

We are now ready to introduce the definition of L-stationarity in MOO.

Definition 8.3. A point $\bar{x} \in \Omega$ is *L-stationary* for Problem (8.1) if $\theta^L(\bar{x}) = 0$.

Remark 8.1. By simple algebraic manipulations, the problem in (8.6) can be rewritten as

$$\min_{z \in \Omega} \max_{j=1,\dots,m} \nabla f_j(\bar{x})^\top (z - \bar{x}) + \frac{L}{2} \|z - \bar{x}\|^2 . \tag{8.7}$$

We can now observe that, if $m = 1$, Definition 8.3 actually coincides with scalar L-stationarity. Indeed, exploiting (8.7), $\theta^L(\bar{x})$ is equivalent to

$$\min_{z \in \Omega} \nabla f(\bar{x})^\top (z - \bar{x}) + \frac{L}{2} \|z - \bar{x}\|^2 = \min_{z \in \Omega} \frac{L}{2} \|z - \bar{x} + \frac{1}{L} \nabla f(\bar{x})\|^2 - \frac{1}{2L} \|\nabla f(\bar{x})\|^2.$$

The minimum in the above problem is attained for $z^* \in \Pi_\Omega[\bar{x} - \frac{1}{L} \nabla f(\bar{x})]$, with Π_Ω being the (not unique) Euclidean projection onto the nonconvex set Ω. We thus have that $\theta^L(\bar{x}) = 0$ if $\frac{L}{2} \|z^* - \bar{x}\|^2 + \nabla f(\bar{x})^\top (z^* - \bar{x}) = 0$, which is satisfied if $\bar{x} \in \Pi_\Omega[\bar{x} - \frac{1}{L} \nabla f(\bar{x})]$, i.e., $\bar{x}$ is L-stationary according to [3].

In the rest of the section, we analyze the relations between L-stationarity, Pareto optimality, Pareto-stationarity and MOLZ conditions. We begin showing that, for any $L > 0$, each L-stationary point is Pareto-stationary.

Proposition 8.1. *Let $\bar{x} \in \Omega$ be an L-stationary point for Problem* (8.1) *with $L > 0$. Then, $\bar{x}$ is Pareto-stationary for* (8.1).

Proof. By contradiction, we assume that $\bar{x}$ is not Pareto-stationary for (8.1), i.e., there exists $\hat{d} \in \mathcal{D}(\bar{x})$ such that

$$0 > \max_{j\in\{1,\dots,m\}} \nabla f_j(\bar{x})^\top \hat{d} + \frac{1}{2}\|\hat{d}\|^2 \geq \max_{j\in\{1,\dots,m\}} \nabla f_j(\bar{x})^\top \hat{d}, \tag{8.8}$$

where the second inequality is justified by the non-negativity of the norm operator.

We now define the direction $\tilde{d}(t) = t\hat{d}$. Given the definition of $\mathcal{D}(\bar{x})$ and the feasibility of $\hat{d}$, we have there exists $\bar{t} > 0$ such that $\bar{x}+\tilde{d}(t) \in \Omega\ \forall t \in [0, \bar{t}]$. Thus, by definition of $\mathcal{D}^L(\bar{x})$, for all $t \in [0, \bar{t}]$, $\tilde{d}(t) \in \mathcal{D}^L(\bar{x})$. Let us define the function $\tilde{\theta}^L : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ as $\tilde{\theta}^L(x, d) = \max_{j=1,\dots,m} \nabla f_j(x)^\top d + \frac{L}{2}\|d\|^2$. By (8.6), it follows that $\theta^L(\bar{x}) = \tilde{\theta}^L(\bar{x}, \bar{v}^L)$, where $\bar{v}^L \in \mathcal{V}^L(\bar{x})$, and also

$$\theta^L(\bar{x}) \leq \tilde{\theta}^L(\bar{x}, d), \quad \forall\, d \in \mathcal{D}^L(\bar{x}). \tag{8.9}$$

Combining the definitions of $\tilde{d}(t)$ and $\tilde{\theta}^L(x, d)$, we get that $\tilde{\theta}^L(\bar{x}, \tilde{d}(t)) = t \max_{j=1,\dots,m} \nabla f_j(\bar{x})^\top \hat{d} + t^2 \frac{L}{2}\|\hat{d}\|^2$. It is easy to see that $\tilde{\theta}^L(\bar{x}, \tilde{d}(t)) < 0$ if

$$0 < t < -\frac{2}{L\|\hat{d}\|^2} \max_{j=1,\dots,m} \nabla f_j(\bar{x})^\top \hat{d}, \tag{8.10}$$

where the right-hand side is a positive quantity as $L > 0$ and (8.8) holds.

Then, taking into account the feasibility of $\tilde{d}(t)$, (8.9) and (8.10), we can define a direction $\tilde{d}(\hat{t})$, with $\hat{t} \in \left(0, \min\{\bar{t}, -\frac{2}{L\|\hat{d}\|^2} \max_{j=1,\dots,m} \nabla f_j(\bar{x})^\top \hat{d}\}\right)$, so that $\tilde{d}(\hat{t}) \in \mathcal{D}^L(\bar{x})$ and $\theta^L(\bar{x}) \leq \tilde{\theta}^L(\bar{x}, \tilde{d}(\hat{t})) < 0$. We finally get a contradiction since, by hypothesis, $\bar{x}$ is an L-stationary point for (8.1), i.e., $\theta^L(\bar{x}) = 0$. □

The last result also highlights the relation between L-stationarity and MOLZ conditions, which, as stated in Lemma 8.3, are necessary for Pareto stationarity. We formalize it in the next corollary.

Corollary 8.1. *Let $\bar{x} \in \Omega$ be an L-stationary point for Problem* (8.1) *with $L > 0$. Then, $\bar{x}$ satisfies the MOLZ conditions.*

Given Proposition 8.1 and Lemma 8.2, we can also state that, under convexity assumptions for $F(\cdot)$, L-stationarity is a sufficient condition for local weak Pareto optimality.

Corollary 8.2. *Assume that $F(\cdot)$ is component-wise convex and let $\bar{x} \in \Omega$ be an L-stationary point for Problem* (8.1) *with $L > 0$. Then, $\bar{x}$ is locally weakly Pareto optimal for* (8.1).

In order to continue the analysis, we need to introduce a couple of notions. The first one is an assumption similar to the one used for L-stationarity in [3], while the second one concerns an adaptation of the *descent lemma* to MOO.

Assumption 8.1. For all $j \in \{1, \ldots, m\}$, $\nabla f_j(\cdot)$ is Lipschitz-continuous over $\mathbb{R}^n$ with constant $L(f_j)$, i.e., $\|\nabla f_j(x) - \nabla f_j(y)\| \leq L(f_j)\|x - y\|$ for all $x, y \in \mathbb{R}^n$.

In what follows, we indicate with $L(F) \in \mathbb{R}^m$ the vector of the Lipschitz constants, i.e., $L(F) = (L(f_1), \ldots, L(f_m))^\top$.

Lemma 8.6 ([5, Proposition A.24])**.** *Let $f_j(\cdot)$, $j = 1, \ldots, m$, be a continuously differentiable function satisfying Assumption 8.1. Then, for all $L \geq L(f_j)$ and any $x, d \in \mathbb{R}^n$, we have that $f_j(x+d) \leq f_j(x) + \nabla f_j(x)^\top d + \frac{L}{2}\|d\|^2$.*

We are ready to show that, for specific L values, the L-stationarity condition is necessary for weak Pareto optimality.

Proposition 8.2. *Let Assumption 8.1 hold, $\bar{x} \in \Omega$ be a weakly Pareto optimal point for Problem* (8.1) *and $L > \max_{j=1,\ldots,m} L(f_j)$. Then, $\bar{x}$ is L-stationary for* (8.1). *Moreover, we have that $\mathcal{V}^L(\bar{x}) = \{\mathbf{0}_n\}$, i.e., the set $\mathcal{V}^L(\bar{x})$ is a singleton.*

Proof. By contradiction, let us assume that either $\bar{x}$ is not L-stationary for (8.1) or $\mathcal{V}^L(\bar{x}) \setminus \{\mathbf{0}_n\} \neq \emptyset$. Then, there exists a direction $\hat{d} \in \mathcal{D}^L(\bar{x})$ such that $\hat{d} \neq \mathbf{0}_n$ and

$$\max_{j \in \{1,\ldots,m\}} \nabla f_j(\bar{x})^\top \hat{d} + \frac{L}{2}\|\hat{d}\|^2 \leq 0. \tag{8.11}$$

By Lemma 8.6, we have that, for all $h \in \{1, \ldots, m\}$,

$$f_h\left(\bar{x} + \hat{d}\right) \leq f_h(\bar{x}) + \nabla f_h(\bar{x})^\top \hat{d} + \frac{L\,(f_h)}{2}\|\hat{d}\|^2. \tag{8.12}$$

From Equation (8.11), we get that $\nabla f_h(\bar{x})^\top \hat{d} \leq \max_{j \in \{1,\ldots,m\}} \nabla f_j(\bar{x})^\top \hat{d} \leq -\frac{L}{2}\|\hat{d}\|^2$, where the first inequality comes from the definition of maximum

operator. Recalling the hypothesis on L and the non-negativity of the norm, we combine (8.12) and the last result obtaining that

$$f_h\left(\bar{x}+\hat{d}\right) \leq f_h(\bar{x}) + \frac{L\left(f_h\right) - L}{2}\|\hat{d}\|^2 < f_h(\bar{x}) + \frac{L\left(f_h\right) - \max_{j\in\{1,\dots,m\}} L\left(f_j\right)}{2}\|\hat{d}\|^2.$$

Thus, for all $h \in \{1,\dots,m\}$ we have $f_h(\bar{x}+\hat{d}) - f_h(\bar{x}) < \frac{\|\hat{d}\|^2}{2}(L(f_h) - \max_{j\in\{1,\dots,m\}} L(f_j)) \leq 0$, leading to the conclusion that we have found a point $\bar{x}+\hat{d} \in \Omega$ such that $F(\bar{x}+\hat{d}) < F(\bar{x})$. This is a contradiction since, by hypothesis, $\bar{x}$ is weakly Pareto optimal for (8.1). Thus, we get the thesis. □

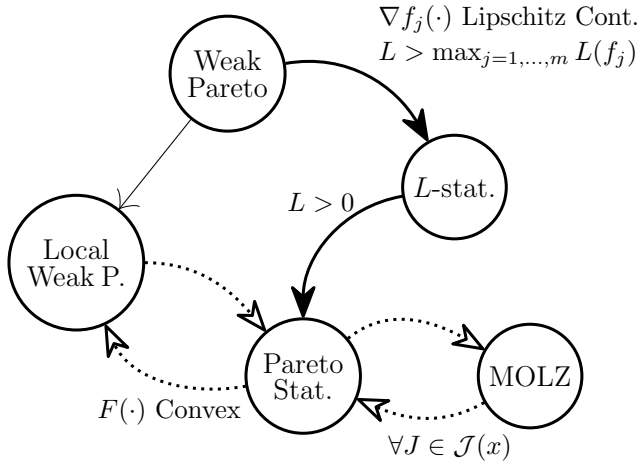


Figure 8.1: Graphical scheme of the theoretical relationships among (local) weak Pareto optimality, L-stationarity, Pareto stationarity and MOLZ conditions. The L-stationarity properties are displayed with solid arrows, while the other ones, stated and proved in [57], are indicated with dotted arrows.

All the theoretical relationships stated in this section are shown in a graphical and compact view in Figure 8.1. The analysis on L-stationarity highlights how the choice of the L value could be crucial: if L is too small, L-stationarity might not be a necessary optimality condition; on the other hand, if L gets too large, all the Pareto stationary points also become L-stationary. This behavior can be better noticed with an example.

Example 8.1. Let us consider the following optimization problem:

$$\min_{x\in\mathbb{R}^2} \quad \frac{1}{2}\left((x_1-3)^2 + (x_2-2.5)^2, (x_1-1)^2 + (x_2-0.5)^2\right)^\top \text{ s.t. } \|x\|_0 \leq 1.$$

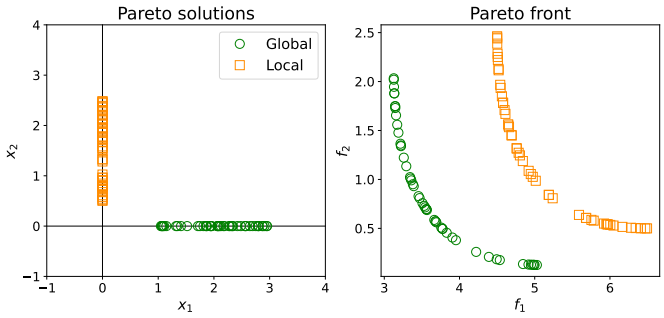
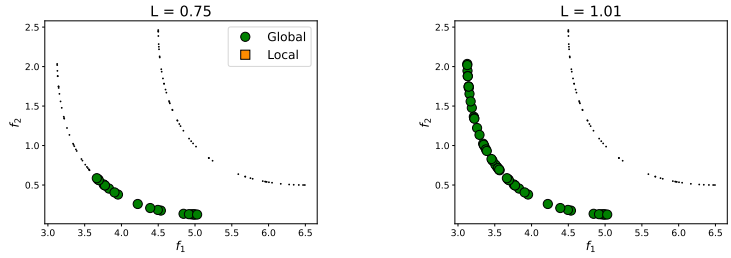


Figure 8.2: Pareto optimal solutions and Pareto front of problem of Example 8.1.

The Lipschitz constant of the gradient of both objective functions $f_j(\cdot)$ is $L(f_j) = 1$. In Figure 8.2, the Pareto optimal solutions and the Pareto front are plotted: the problem has global optimal solutions corresponding to points with $x_1 \neq 0$; the local ones are characterized by the second component $x_2 \neq 0$. By Lemmas 8.1-8.3, it follows that all the considered points are Pareto-stationary and satisfy the MOLZ conditions. In Figure 8.3, we show which Pareto solutions are L-stationary, considering four different choices for L. If L is chosen too small (Figure 8.3a), some global Pareto solutions do not result to be L-stationary. As stated in Proposition 8.2, the L-stationarity condition turns out to be necessary for Pareto optimality for an L value greater than the Lipschitz constants (Figure 8.3b where $L = 1.01$). On the other hand, a too high value makes the condition rather weak: in Figure 8.3c ($L = 1.25$), even some local Pareto solutions are L-stationary. The situation is further stressed in Figure 8.3d where $L = 2$ and all Pareto-stationary points are also L-stationary.

8.3 New Algorithmic Approaches for Sparse MOO Problems

In this section, we propose two procedures to solve cardinality-constrained MOO problems. The first one can be seen as an extension of the *Iterative Hard Thresholding* (`IHT`) algorithm [3]; the second one is a front approximation approach that takes as input candidate solutions, possibly associated



(a) $L = 0.75$ (b) $L = 1.01$

(c) $L = 1.25$ (d) $L = 2$

Figure 8.3: L-stationary points in the Pareto front of problem of Example 8.1 ($L(F) = (1, 1)^\top$) for different values of L.

with different support sets, and then spans the portions of the Pareto front associated with those supports. We report their schemes and discuss their properties in separate sections.

8.3.1 Multi-Objective Iterative Hard Thresholding

The first procedure we introduce is the *Multi-Objective Iterative Hard Thresholding* (`MOIHT`) algorithm. The scheme of the method is reported in Algorithm 8.1. At each iteration of `MOIHT`, the current solution x_k is updated solving Problem (8.13). The execution continues until an L-stationary point for (8.1) is found.

Remark 8.2. At each iteration k, the solution x_{k+1} generated by `MOIHT` is feasible for (8.1). Indeed, the feasibility easily follows by definition of $\mathcal{D}^L(x_k)$.

Remark 8.3. It is very important to underline that Step 4 is a practical operation that can be effectively implemented in the general case. Problem (8.13) can indeed be solved up to global optimality, for example with mixed-integer programming techniques (see, e.g., [8,9]). Defining a sufficiently large scalar $M > 0$, the problem can be equivalently reformulated as

$$\min_{\beta,d,\delta} \ \beta + \frac{L}{2}\|d\|^2 \quad \text{s.t.} \quad \nabla f_j(\bar{x})^\top d \leq \beta \quad \forall j = 1,\dots,m, \qquad \mathbf{1}_n^\top \delta \leq s,$$
$$-M\delta \leq \bar{x} + d \leq M\delta, \qquad \beta \in \mathbb{R},\ d \in \mathbb{R}^n,\ \delta \in \{0,1\}^n.$$

Algorithm 8.1: Multi-Objective Iterative Hard Thresholding (`MOIHT`)

1 Input: $F : \mathbb{R}^n \to \mathbb{R}^m$, Ω feasible set, $x_0 \in \Omega$, $L > \max_{j\in\{1,\dots,m\}} L(f_j)$.
2 $k = 0$
3 **while** x_k *is not L-stationary for Problem* (8.1) **do**
4 Compute

$$\mathcal{V}^L(x_k) = \operatorname*{arg\,min}_{d\in\mathcal{D}^L(x_k)} \max_{j\in\{1,\dots,m\}} \nabla f_j(x_k)^\top d + \frac{L}{2}\|d\|^2 \qquad (8.13)$$

5 Let $v_k^L \in \mathcal{V}^L(x_k)$
6 $x_{k+1} = x_k + v_k^L$
7 Let $k = k + 1$
8 **return** x_k

8.3.1.1 Convergence Analysis

In this section, we provide a detailed theoretical analysis of `MOIHT`, where we suppose that Assumption 3.1 holds.

Before proving the main convergence result, we need to prove a technical lemma.

Lemma 8.7. *Let Assumptions 3.1-8.1 hold and $\{x_k\}$ be the sequence generated by Algorithm 8.1 with constant $L > \max_{j\in\{1,\dots,m\}} L(f_j)$. Then:*

1. *for all k, $F(x_k) - F(x_{k+1}) \geq \frac{1}{2}\|x_k - x_{k+1}\|^2(\mathbf{1}_m L - L(F))$;*

2. *for all* k, *if* $x_k \neq x_{k+1}$, *then* $F(x_{k+1}) < F(x_k)$;

3. *for all* $j \in \{1, \ldots, m\}$, *the sequence* $\{f_j(x_k)\}$ *is non-increasing;*

4. *the sequence* $\{F(x_k)\}$ *converges;*

5. $\lim_{k\to\infty} \|x_k - x_{k+1}\|^2 = 0$.

Proof. See Appendix C. □

Proposition 8.3. *Let Assumptions 3.1-8.1 hold and* $\{x_k\}$ *be the sequence generated by Algorithm 8.1 with constant* $L > \max_{j\in\{1,\ldots,m\}} L(f_j)$. *Then, the sequence admits cluster points, each one being* L*-stationary for Problem* (8.1).

Proof. First, we prove that the sequence admits limit points. By Lemma 8.7, we deduce that, for all k, $F(x_k) \leq F(x_{k-1}) \leq \ldots \leq F(x_0)$. Moreover, as noted in Remark 8.2, $x_k \in \Omega$ for all k. Thus, we have that $x_k \in \mathcal{L}_F(F(x_0))\ \forall k$. Since Assumption 3.1 holds, we conclude that the sequence $\{x_k\}$ is bounded and it thus admits limit points.

Now, we denote as $\bar{x}$ a limit point, i.e., there exists a subsequence $K \subseteq \{0, 1, \ldots\}$ such that

$$\lim_{\substack{k\to\infty \\ k\in K}} x_k = \bar{x}.$$

By Point 5. of Lemma 8.7 and Step 6, we have that $\lim_{k\to\infty, k\in K} \|v_k^L\|^2 = 0$ and, thus, $v_k^L \to_K \mathbf{0}_n$. Considering this last result and taking the limit for $k \to \infty, k \in K$, in Problem (8.13), by Point 3. of Lemma 8.5, Steps 4-5, the continuously differentiability of $F(\cdot)$ and the continuity of the maximum and norm operators, we get that

$$\theta^L(\bar{x}) = \lim_{\substack{k\to\infty \\ k\in K}} \theta^L(x_k) = \lim_{\substack{k\to\infty \\ k\in K}} \max_{j\in\{1,\ldots,m\}} \nabla f_j(x_k)^\top v_k^L + \frac{L}{2}\|v_k^L\|^2 = 0.$$

We conclude that $\theta^L(\bar{x}) = 0$ and, then, $\bar{x}$ is L-stationary for Problem (8.1). □

Remark 8.4. By Proposition 8.3 and the continuity of $\theta^L(\cdot)$ (Lemma 8.5) we are guaranteed that, for any $\varepsilon > 0$, Algorithm 8.1 will produce a point x_k such that $\theta^L(x_k) > -\varepsilon$ in a finite number of iterations. Thus, we can effectively employ this condition as a practical stopping criterion for the `MOIHT` procedure. A similar finite termination property is also proved for the general framework for single-point methods (Algorithm 3.1) in Proposition 3.1. However, in that result Ω is assumed to be a convex feasible set.

8.3.2 Sparse Front Steepest Descent

In what follows, we describe and analyze the *Sparse Front Steepest Descent* (`SFSD`) methodology. The algorithm can be seen as a two phases approach, which is based on the following consideration: in problems of the form (8.1), the Pareto front is usually an irregular set made up of several, distinct smooth parts; each of these nice portions of the front is typically the image of a set of solutions sharing the same structure, i.e., associated with the same support set. The rationale of the proposed algorithm is thus to first define a set of starting solutions; the support sets of these solutions should ideally be diverse and define a subspace where a portion of the Pareto set lies. Then, an adaptation of the *Front Steepest Descent* (`FSD`) algorithm (Section 3.3) can be run starting from this initial set of solutions to span the front exhaustively. To the best of our knowledge, `SFSD` is the first front-oriented approach for cardinality-constrained MOO.

8.3.2.1 Phase One: Initialization

The first phase of the `SFSD` procedure deals with the identification of a set of starting solutions. The most direct way of proceeding would arguably be exhaustive enumeration of the super support sets, selecting for each a solution. However, the number of possible supports is high, growing as fast as $\binom{n}{s}$, but only a small fraction contributes to the Pareto front. Thus, this strategy is inefficient, up to being totally impractical with problems of nontrivial size.

A totally random initialization might also appear to be a possible path to take, but, by similar reasons as above, it would end up being a completely luck-based operation. Therefore, we suggest to exploit single-point solvers to retrieve Pareto-stationary solutions. Indeed, by the mechanisms of this kind of algorithms, not only the obtained points are stationary but are usually also good solutions from a global optimization perspective. In the numerical experiments (Section 9.6), we explored the following (not exhaustive) list of options.

- Using the `MOIHT` discussed in Section 8.3.1 in a multi-start fashion. Since the algorithm finds L-stationary solutions, optimization should be driven avoiding "bad" supports.
- Using the *Multi-Objective Sparse Penalty Decomposition* (`MOSPD`) method from [57] in a multi-start fashion; in brief, at each iteration k of `MOSPD`,

a pair (x_{k+1}, y_{k+1}) is found such that x_{k+1} is (approximately) Pareto-stationary for the penalty function $Q_{\tau_k}(x, y_{k+1}) = F(x) + \frac{\tau_k}{2}\mathbf{1}_m\|x - y_{k+1}\|^2$, with $\tau_k \to \infty$ for $k \to \infty$. The pair (x_{k+1}, y_{k+1}) is obtained by means of an *alternate minimization* scheme. For further details, we refer the reader to [57]. `MOSPD` is proved to converge to points satisfying the multi-objective Lu-Zhang conditions for Problem (8.1), that are even weaker than Pareto-stationarity; however, Penalty Decomposition methods have been shown to retrieve solutions both in the scalar [53] and in the multi-objective [57] case that are good from a global optimization perspective.

- Combining the strategies of the two preceding points: for each point of a multi-start random initialization, we can first run the `MOSPD` procedure to exploit its exploration capabilities; then, we can use `MOIHT` in cascade, so that bad Lu-Zhang points that are not L-stationary can eventually be escaped. We refer to this approach as `MOHyb`.

- Solving the scalarized - single objective - problem for different trade-off parameters.

Once the starting set of solutions is obtained by one of the above strategies, a further step has to be carried out. Indeed, we need to associate each solution with a super support set. Now, if a solution has full support, then there is a unique super support and no ambiguity. However, there might be solutions with incomplete support; these solutions might be not Pareto-stationary (for example, if obtained with `MOSPD`), in which case we shall carry out a descent step along the steepest feasible descent direction; if on the other hand we actually have a Pareto-stationary point with incomplete support, we shall associate it with any of the super supports.

Obviously, we can complete this first phase with a filtering operation, where dominated solutions get discarded. To sum up, the result of the first phase of the algorithm provides a set of starting solutions each one associated with a super support set.

8.3.2.2 Phase Two: Front Steepest Descent Adaptation

In Algorithm 8.2, we report the scheme of the proposed algorithmic framework `SFSD`.

Algorithm 8.2: Sparse Front Steepest Descent (`SFSD`)

Input: $F : \mathbb{R}^n \to \mathbb{R}^m$, Ω feasible set, $\alpha_0 > 0$, $\delta \in (0,1)$.
$\mathcal{X}^0 = \texttt{Initialize}(F(\cdot))$
$k = 0$
while *a stopping criterion is not satisfied* **do**
  $\widehat{\mathcal{X}}^k = \mathcal{X}^k$
  forall $(x_c, J_{x_c}) \in \mathcal{X}^k$ **do**
    if $(x_c, J_{x_c}) \in \widehat{\mathcal{X}}^k$ **then**
      if $\theta^{J_{x_c}}(x_c) < 0$ **then**
        $\alpha_c^k = \texttt{ALS}(F(\cdot), \Omega, x_c, v^{J_{x_c}}(x_c))$
      else
        $\alpha_c^k = 0$
      $z_c^k = x_c + \alpha_c^k v^{J_{x_c}}(x_c)$
      $\widehat{\mathcal{X}}^k = \left(\widehat{\mathcal{X}}^k \cup \left\{(z_c^k, J_{x_c})\right\}\right) \setminus \left\{(y, J_y) \in \widehat{\mathcal{X}}^k \mid J_y = J_{x_c},\ F(z_c^k) \lneqq F(y)\right\}$
      forall $\mathcal{I} \subseteq \{1, \ldots, m\}$ *s.t.* $\theta_{\mathcal{I}}^{J_{x_c}}(z_c^k) < 0$ **do**
        if $(z_c^k, J_{x_c}) \in \widehat{\mathcal{X}}^k$ **then**
          $\alpha_c^{\mathcal{I}} = \max_{h \in \mathbb{N}} \{\alpha_0 \delta^h \mid \forall\, (y, J_y) \in \widehat{\mathcal{X}}^k,\ J_y = J_{x_c},\ \exists j \in \{1, \ldots, m\} \text{ s.t. } f_j(z_c^k + \alpha_0 \delta^h v_{\mathcal{I}}^{J_{x_c}}(z_c^k)) < f_j(y)\}$
          $\hat{z} = z_c^k + \alpha_c^{\mathcal{I}} v_{\mathcal{I}}^{J_{x_c}}(z_c^k)$
          $\widehat{\mathcal{X}}^k = \left(\widehat{\mathcal{X}}^k \cup \{(\hat{z}, J_{x_c})\}\right) \setminus \left\{(y, J_y) \in \widehat{\mathcal{X}}^k \mid J_y = J_{x_c},\ F(\hat{z}) \lneqq F(y)\right\}$
  $\mathcal{X}^{k+1} = \widehat{\mathcal{X}}^k$
  $k = k + 1$
return $\mathcal{X}^k$

The method starts working with the starting set of solutions resulting from the `Initialize` step, i.e., phase one of the algorithm; the obtained set $\mathcal{X}^0$ is then given by

$$\mathcal{X}^0 = \{(x, J_x) \mid J_x \in \mathcal{J}(x)\},$$

i.e., solutions associated with a corresponding super support set. Given any pairs $(x, J_x), (y, J_y) \in \mathcal{X}^0$ with $J_x = J_y$, we assume that x and y are mutually

nondominated w.r.t. $F(\cdot)$.

Basically, the SFSD algorithm employs the instructions of the *Front Steepest Descent* (FSD) algorithm (Section 3.3), modified as suggested in Chapter 6, treating separately points associated with different super support sets.

Specifically, for any nondominated point x_c in the current Pareto front approximation, a common descent step in the subspace corresponding to the support J_{x_c} is carried out, doing a standard Armijo-type line search (Algorithm 3.2). In other words, the search direction is thus given by $v^{J_{x_c}}(x_c)$ according to (8.5). Then, further searches w.r.t. subsets of objectives are carried out from the obtained point, as long as it is not dominated by any other point y in the set with $J_y = J_{x_c}$. These additional explorations are carried out along partial descent directions in the reference subspace of the point at hand. Considering $\mathcal{I} \subseteq \{1, \ldots, m\}$ as a subset of objectives indices, we define $\theta^J_{\mathcal{I}}(\bar{x}) = \min_{d \in \mathbb{R}^n} \max_{j \in \mathcal{I}} \nabla f_j(\bar{x})^\top d + \frac{1}{2}\|d\|^2$ *s.t.* $d_{\bar{J}} = \mathbf{0}_{|\bar{J}|}$. Similar to (8.5), the problem has a unique solution that we denote by $v^J_{\mathcal{I}}(\bar{x})$.

Since the solutions are compared only if associated to the same super support set, the subspaces induced by different super support sets are explored separately in SFSD. As a result, we basically obtain separate Pareto front approximations, each one corresponding to a super support set. At the end of the SFSD execution, all the points can be compared and the dominated ones can finally be filtered out in order to obtain the final Pareto front approximation for Problem (8.1).

Note that, conceptually, the algorithm can be seen as if multiple, independent runs of the IFSD method (Chapter 6) were carried out, each time constraining the optimization process in a particular subspace; however, exploration in SFSD is carried out "in parallel" throughout different supports, so that the front approximation is constructed uniformly and we can avoid cases where all the computational budget is spent for the optimization w.r.t. the first few considered supports.

Remark 8.5. Since each point is considered for search steps only in the subspace induced by its associated super support set, it easily follows that every new solution will be feasible for (8.1).

8.3.2.3 Algorithm Theoretical Analysis

In this section, we state the convergence property of the SFSD methodology. We refer the reader to Chapter 6 for the proofs of properties inherited by SFSD directly from the IFSD method.

Before proving the convergence result, we need to introduce the set $X_J^k = \left\{x \mid \exists\,(x, J) \in \mathcal{X}^k\right\}$, with J denoting a super support set.

Remark 8.6. The SFSD methodology also inherits the well-defineteness property from the front steepest descent method. In particular, the soundness of the line searches holds by Propositions 6.1, 6.2 and 6.3 stated in Chapter 6. In fact, proofs can be adapted easily, taking into account that SFSD deals separately with one or multiple sets, each corresponding to a different super support set J.

In the next proposition, we make use of both Point 2. of Assumption 3.2 and the concept of linked sequence (Definition 3.1).

Proposition 8.4. *Let Point 2. of Assumption 3.2 hold with X_J^0, a set of mutually nondominated points associated with the super support set J, and a point $x_0^J \in X_J^0$. Let $\left\{X_J^k\right\}$ be the sequence of sets of nondominated points produced by Algorithm 8.2. Let $\left\{x_{j_k}^J\right\}$ be a linked sequence, then it admits accumulation points, each one satisfying the MOLZ conditions for Problem* (8.1).

Proof. By the instructions of the algorithm, each linked sequence $\left\{x_{j_k}^J\right\}$ can be seen as a linked sequence generated by applying the IFSD algorithm (Chapter 6) to the problem of minimizing $F(x)$ subject to $x_{\bar{J}} = \mathbf{0}_{|\bar{J}|}$. Thus, we can follow the proof of Proposition 6.4 to show that each accumulation point $\bar{x}$ of the linked sequence $\left\{x_{j_k}^J\right\}$ is such that $\theta^J(\bar{x}) = 0$, i.e., $\bar{x}$ satisfies the MOLZ conditions for (8.1). □

8.4 Conclusions

In this chapter, we considered cardinality-constrained multi-objective optimization problems. Inspired by the homonymous condition for sparse SOO [3], we defined the L-stationarity concept for MOO and we analyzed its relationships with the main Pareto optimality concepts and conditions.

Then, we proposed two novel algorithms for the considered class of problems. The first one is an extension of the *Iterative Hard Thresholding* method [3] to the MOO case, called MOIHT: like the original approach, it aims to generate an L-stationary point. The second algorithm called *Sparse Front Steepest Descent* (SFSD) is, to the best of our knowledge, the first front-oriented approach for cardinality-constrained MOO. Being an adaptation of the front

steepest algorithm [23], SFSD aims to approximate the (typically irregular and fragmented) Pareto front of the problem at hand. The method depends on suitable initialization strategies, including, e.g., multi-starting the MOIHT or the MOSPD [57] algorithms, an hybridization of the two, or a scalarization approach. From a theoretical point of view, we proved for MOIHT that the sequence of points converges to L-stationary solutions; for SFSD, on the other hand, we stated global convergence to points satisfying the MO Lu-Zhang optimality conditions.

By a numerical experimentation (Section 9.6), we evaluated the performance of the proposed methodologies on benchmarks of quadratic and logistic regression problems. The SFSD methodology is thus shown to be successful at spanning the Pareto front in an exhaustive way, with the multi-start hybrid MOSPD-MOIHT procedure (MOHyb) being the most promising solution to be used in the first phase of the algorithm.

Chapter 9

Computational Experiments

In this chapter, we provide the results of thorough computational experiments to evaluate the efficiency and effectiveness of the algorithms proposed in the thesis. The full code of the experiments was written in Python3. In addition, all the tests were run on a computer with the following characteristics: Ubuntu 22.04, Intel Xeon Processor E5-2430 v2 6 cores 2.50 GHz, 16 GB RAM. In order to solve instances of the search direction problem (2.3), we employed the Gurobi Optimizer (Version 9.1) [48].

9.1 Experimental Settings

In this section, we report detailed information on the metrics, the settings used for the main tested algorithms and, finally, the problems used to carry out the comparisons.

9.1.1 Metrics

In this section, we provide a little description of the main metrics and tools used to compare the algorithms.

The first three metrics are the ones introduced in [25], which are very popular and used by the multi-objective optimization community: *Purity*, Γ*–spread* and Δ*–spread*.

We recall that the *Purity* metric measures the quality of the generated front, i.e., how effective a solver is at obtaining non-dominated points w.r.t. its competitors. In detail, the *Purity* value indicates the ratio of the number

of non-dominated points that a solver obtained over the number of the points produced by that solver. Clearly, a higher value is related to a better performance. In order to calculate the *Purity* metric, we need a reference front to establish whether a point is dominated or not. In our experiments, we considered as the reference front the one obtained by combining the fronts retrieved by the tested algorithms and by discarding the dominated points.

The *spread* metrics are equally essential, since they measure the uniformity of the generated fronts in the objectives space. In particular, the Γ*–spread* is defined as the maximum ℓ_∞ distance in the objectives space between adjacent points of the Pareto front (extremes of the reference front included), while the Δ*–spread* is quite similar to the standard deviation of the ℓ_∞ distance. As opposed to the *Purity*, low values for the *spread* metrics are associated with good performance.

The fourth metric we employed is the *Hypervolume* (HV) [110]: it calculates the area/volume which is dominated by the provided set of solutions with respect to a reference point. The latter is chosen such that the value of each of its coordinates is slightly greater than the worst value obtained by any of the compared solvers on the related objective function. Similar to *Purity*, higher values for the *Hypervolume* metric mean better performance.

Lastly, we employed the performance profiles introduced in [27], that are an useful tool to better appreciate the relative performance and robustness of the tested algorithms. The performance profile of a solver w.r.t. a certain metric is the (cumulative) distribution function of the ratio of the score obtained by the solver over the best score among those achieved by all the considered solvers. In other words, it is the probability that the metric score achieved by the solver in a problem is within a factor $\tau \in \mathbb{R}$ of the best value obtained by any of the solvers in that problem. For a more technical explanation about performance profiles, we refer the reader to [27]. Note that performance profiles w.r.t. *Purity* and *Hypervolume* were produced based on the inverse of the obtained values, since these metrics have increasing values for better solutions.

9.1.2 Algorithms and Hyper-parameters

The main front-oriented algorithms we tested for the comparisons are the following.

- The *Front Steepest Descent* (`FSD`) algorithm [23], whose description

can be found in Section 3.3, is the representative for front-oriented descent methods for unconstrained MOO problems. We also tested FSD equipped with the extrapolation strategy proposed in [23] and mentioned in Remark 3.3: this variant is indicated as EFSD.

- The *Front Projected Gradient* (FPG) method. In Appendix A, the reader can find a description and a theoretical analysis of this approach. In brief, it is a variant of FSD capable of handling box-constrained MOO problems.

- NSGA-II [26], which we describe in detail in Section 4.1, is the most popular evolutionary algorithm for both unconstrained and box-constrained MOO problems. The parameters values for NSGA-II were chosen according to the referenced paper.

- The *Direct Multi-Search* (DMS) algorithm [25], which is a multi-objective derivative-free method, inspired by the search/poll paradigm of direct-search methodologies of directional type. The parameters for this method were set according to the referenced paper and the code available online (http://www.mat.uc.pt/dms).

- The MOSQP algorithm [37], which we consider a gradient-based SQP–type competitor for convex constrained nonlinear MOO problems. The chosen hyper-parameters for MOSQP were the best ones according to [37]. For the quadratic approximations, we used $H_i = I_n$ in the second stage and $H_i = \nabla^2 f_i(x_k) + E_i$ (E_i being obtained by a modified Cholesky algorithm) in the third stage, as it is claimed to be the most robust and efficient way to use MOSQP [37]. For a more detailed explanation about the various MOSQP stages and versions, we refer to [37]. Lastly, we used the Ipopt software package (https://github.com/coin-or/Ipopt) [102] in order to solve the SQP problems.

As for the hyper-parameters of our approaches, they were chosen based on some preliminary experiments on subsets of the tested problems, which we do not report here for the sake of brevity.

- For NSMA (Section 4.2), we set $N = 100$, $s_h = 10$, $q = 0.9$, $n_{opt} = 5$. The same number N of solutions in the population was also used for NSGA-II.

- For `FRONT-ALAMO` (Section 7.1), we set: $\tau_0 = 1$; if the problem only has bound constraints $\rho = 10$, otherwise $\rho = 2$; $\sigma = 0.9$; $\overline{\mu} = 10^4$; $\mu^0 = \mathbf{0}_p$.
- For all the Armijo-Type Line Searches, we set $\alpha_0 = 1$, $\delta = 0.5$, $\gamma = 10^{-4}$.
- For `IFSD` (Section 6.2), `FRONT-ALAMO` and `SFSD` (Section 8.3.2), a strategy to limit the number of points used for partial descent searches was employed, in order to improve the efficiency of the overall procedures and avoid the production of too many, very close solutions. In particular, we added a condition based on the crowding distance [26], of which a brief description can be found in Section 4.1.1, to decide whether a point should be considered for further exploration.

Unless explicitly stated otherwise, for each algorithm and problem the test was run for up to 2 minutes. A stopping criterion based on a time limit is the fairest way to compare such structurally different algorithms. Clearly, we also took into account other specific stopping criteria indicating that a certain algorithm cannot improve the solutions anymore.

`NSMA` and `NSGA-II` are non-deterministic algorithm w.r.t. the others. Therefore, we decided to run them 5 times on every problem, with different seeds for the pseudo-random number generator. Every execution was characterized by the same time limit (2 minutes). The five generated fronts were compared based on the *Purity* metric and only the best one was chosen as the output of `NSMA`/`NSGA-II`. In this context, the reference front was the combination of the fronts of the 5 executions. Executing 5 runs lets `NSMA`/`NSGA-II` reduce its sensibility to the seed used for its random operations. All the other algorithms are deterministic and, then, they were executed once.

9.1.3 Problems

The problems constituting the benchmark of the computational experiments are listed in Table 9.1. In this benchmark, we considered problems whose objective functions are at least continuously differentiable almost everywhere. If a problem is characterized by singularities, we counted these latter ones as Pareto-stationary points. Unless explicitly stated otherwise, the values for n considered for the experiments are the ones shown in the table.

In the table we explicitly divide the problems with at most only box constraints, the ones with general convex constraints and the two new box-constrained convex MOO problems MAN_1 and MAN_2, whose formulation

Source	Problem	m	n	Box	n_l	n_{nl}
[107]	CEC09_1, CEC09_2 CEC09_3, CEC09_4 CEC09_5, CEC09_6 CEC09_7	2	5, 10, 20, 30, 40, 50, 100, 200	✓	N/A	N/A
	CEC09_8, CEC09_9 CEC09_10	3				
[109]	ZDT_1, ZDT_2 ZDT_3, ZDT_4	2	2, 5, 10, 20, 30, 40, 50, 100, 200			
[50], App. D	MOP_2, M-MOP_2					
[80]	MMR_5					
App. D	M-FDS_1	3				
[50]	MOP_1	2	1			
	MOP_3	2	2			
	MOP_7	3				
[51]	JOS_1	2	2, 5, 10, 20, 30 40, 50, 100, 200	✗		
[92]	SLC_2					
[21]	M-BNH_1, M-BNH_2	2	2	✗	N/A	2
	LAP_1	2	2	✗	1	2
	LAP_2	2	2, 5, 10, 20 30, 40, 50 100, 200	✗	N/A	1
App. D	M-OSY	2	6	✓	4	2
App. D	MAN_1	2	2, 5, 10, 20, 30 40, 50, 100, 200	✓	N/A	N/A
	MAN_2	3				

Table 9.1: Problems used in the computational experiments. **Box** indicates if the considered problem has boundary conditions. $\mathbf{n_l}$ indicates the number of linear constraints, without considering the boundary ones, if any. $\mathbf{n_{nl}}$ indicates the number of non linear constraints.

can be found in Appendix D. Regarding the first category, the set is mainly composed by problems with particularly difficult objective functions, such as the CEC09 problems [107] and the ZDT ones [109]. The CEC09 problems have non-continuously differentiable objective functions; the same feature is present in the ZDT problems, where the objective functions are also composite. Hence, these problem classes are particularly interesting for the analysis of the behavior of the algorithms with hard tasks. Note that some problem names are characterized by the prefix *M-*: these problems are rescaled ver-

sions of the original ones and their formulations are provided in Appendix D. Finally, we included in our benchmark the slightly modified versions of the BNH problems and the LAP problems proposed in [21]. We also considered a modification of the OSY problem [85], whose formulation can be found in Appendix D.

Unless explicitly stated otherwise, similar to what is done in [25], for each box-constrained problem, the initial points were uniformly selected from the hyper-diagonal defined by the bound constraints. In this scenario, the number of initial points is equal to the dimension n of the problem. Since in the MOP_1 problem $n = 1$, only in this case we started the tests from one feasible point, namely, $x = 0$.

9.2 Performance Evaluation of the Non-dominated Sorting Memetic Algorithm

In this section [1], we focus on the comparison of `NSMA` (Chapter 4, Section 4.2) with some of the main state-of-the-art methods in diverse settings. In particular, we tested as competitors `FPG`, `NSGA-II` and `DMS`. We consider the first two methods, described in Appendix A and Section 4.1 respectively, the representatives for descent methods and EAs for box-constrained MOO problems and, thus, `NSMA` most direct competitors.

9.2.1 Experimental Comparisons between `NSGA-II` and `FPG`

Before turning to the evaluation of the `NSMA`, we carry out a preliminary study. Evolutionary algorithms and descent methods have their own drawbacks. In particular, EAs do not have theoretical convergence properties. In addition, they can be very expensive in particular settings. On the other side, descent algorithms suffer on highly non-convex problems: in these cases, they often produce sub-optimal solutions, especially when the starting points are not chosen carefully. In this section, we want to address two topics:

- the impact of convexity of the objective functions on the performance of these algorithms;

[1] The implementation code of the `NSMA` algorithm can be found at `https://github.com/pierlumanzu/nsma` [74].

- the behavior of the methods as the problem dimension n increases.

For the comparisons of this section, we only considered the `NSGA-II` and `FPG` algorithms, which we respectively pick as representatives for the two classes of methods. As benchmark, we picked four problems that are scalable w.r.t. the problem dimension n and have the following features: the MAN_1 problem and the ZDT_1 problem have convex objective functions; the CEC09_1 problem and the CEC09_4 problem have nonconvex objective functions. They can be found listed in Table 9.1. For these comparisons, each problem was tested for values of $n \in \{5, 10, 20, 30, 40, 50, 100, 200\}$.

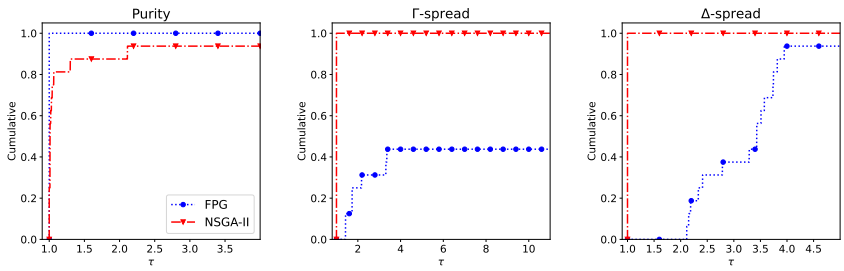


Figure 9.1: Performance profiles for `FPG` and `NSGA-II` on the *convex* MAN_1 and ZDT_1 problems.

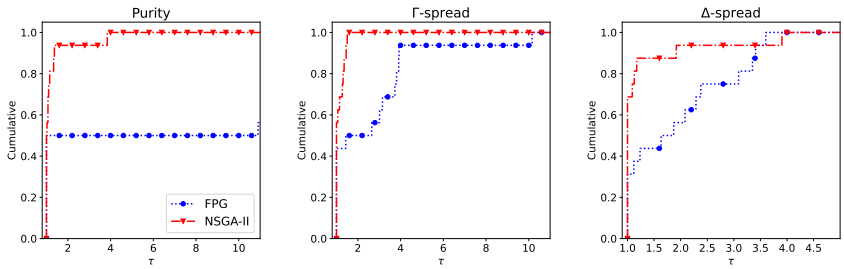


Figure 9.2: Performance profiles for `FPG` and `NSGA-II` on the *nonconvex* CEC09_1 and CEC09_4 problems.

We show the performance profiles for the two algorithms on the convex problems in Figure 9.1 and on the nonconvex problems in Figure 9.2.

n	MAN_1 (F convex)					
	Purity		Γ–*spread*		Δ–*spread*	
	FPG	NSGA-II	FPG	NSGA-II	FPG	NSGA-II
5	**1.0**	**1.0**	55.0	**2.439**	N/A	**0.655**
10	**1.0**	0.97	83.969	**25.021**	1.882	**0.549**
20	**1.0**	0.96	724.921	**218.945**	1.734	**0.623**
30	**1.0**	0.94	3485.981	**1605.224**	1.694	**0.702**
40	**1.0**	0.77	5292.454	**3722.987**	1.863	**0.799**
50	**1.0**	0.474	10330.682	**5964.774**	1.987	**0.924**
100	**1.0**	0.0	82996.647	**58239.443**	1.987	**0.94**
200	**1.0**	**1.0**	1171289.94	**677249.587**	1.964	**0.898**

Table 9.2: Metrics values obtained by FPG and NSGA-II in the MAN_1 problem with $n \in \{5, 10, 20, 30, 40, 50, 100, 200\}$. The metric values marked in bold are the best obtained in the considered instance.

We can observe that in the former case the FPG turned out to be better than NSGA-II in terms of *Purity*. This result reasonably comes from the fact that, in problems characterized by convex objective functions, the use of first-order information and common descent directions lets FPG find better solutions than NSGA-II for equal computational budget. On the contrary, the FPG algorithm was outperformed by NSGA-II in terms of Γ–*spread* and Δ–*spread*. In this perspective, the crossover and mutation operations of NSGA-II allow to consistently obtain spread Pareto front approximations, while the constrained steepest partial descent directions and the B-FALS employed by FPG, whose description can be found in Appendix A, are apparently not as effective.

As for the nonconvex case, we can observe from the *Purity* profile that now the FPG obtained many points that are dominated by those produced by NSGA-II. The results with the *spread* metrics are instead analogous to the convex case, with NSGA-II outperforming FPG. However, the performance gaps in terms of Γ–*spread* and Δ–*spread* are less marked than the ones in the convex case.

In order to assess the performance of the algorithms as the problem dimension n increases, in Tables 9.2-9.3-9.4 we show in detail the metrics values achieved by the two methods on the convex MAN_1 problem and the non-convex CEC09_1 and CEC09_4 problems.

The *Purity* values indicate some relevant features of the two algorithms.

n	CEC09_1 (F non-convex)					
	Purity		Γ*–spread*		Δ*–spread*	
	FPG	NSGA-II	FPG	NSGA-II	FPG	NSGA-II
5	**0.983**	0.92	0.555	**0.149**	1.754	**0.843**
10	**0.982**	0.92	0.555	**0.387**	1.701	**1.042**
20	**1.0**	0.91	**0.327**	0.475	1.971	**1.597**
30	**1.0**	0.75	**0.462**	0.605	**1.578**	1.719
40	**1.0**	0.73	**0.437**	0.593	**1.519**	1.713
50	**1.0**	0.87	**0.402**	0.604	**1.474**	1.729
100	**1.0**	0.978	**0.466**	0.595	**0.427**	1.668
200	**1.0**	0.26	**0.501**	0.571	**0.821**	1.574

Table 9.3: Metrics values obtained by FPG and NSGA-II in the CEC09_1 problem with $n \in \{5, 10, 20, 30, 40, 50, 100, 200\}$. The metric values marked in bold are the best obtained in the considered instance.

n	CEC09_4 (F non-convex)					
	Purity		Γ*–spread*		Δ*–spread*	
	FPG	NSGA-II	FPG	NSGA-II	FPG	NSGA-II
5	0.025	**0.96**	0.421	**0.134**	1.876	**0.789**
10	0.003	**0.97**	0.3	**0.03**	1.972	**0.548**
20	0.001	**0.98**	0.339	**0.087**	1.926	**0.575**
30	0.001	**0.98**	0.395	**0.1**	1.942	**0.57**
40	0.002	**0.98**	0.4	**0.133**	1.864	**0.603**
50	0.02	**0.98**	0.44	**0.115**	1.337	**0.584**
100	0.01	**0.99**	0.384	**0.145**	1.274	**0.682**
200	0.09	**0.98**	**0.451**	0.478	1.208	**1.08**

Table 9.4: Metrics values obtained by FPG and NSGA-II in the CEC09_4 problem with $n \in \{5, 10, 20, 30, 40, 50, 100, 200\}$. The metric values marked in bold are the best obtained in the considered instance.

While in the convex case, FPG outperformed NSGA-II, in the non-convex case, the performance depends on the problem at hand. In the CEC09_1 case, FPG turned out to be capable of obtaining better points than NSGA-II. Moreover, as the value of n increased, the superiority of FPG on the *Purity* metric became even marked. A similar situation can be also seen on the MAN_1 problem, where, except for the $n = 200$ case, the NSGA-II performance degraded on the largest instances. These results highlight one of the drawbacks of the EAs, i.e., the limited scalability. In this case, common descent directions can be very helpful for cheaply improving the quality of the solutions. On the other hand, on problems with more difficult non-convex objective functions, such as the CEC09_4 problem, FPG failed to escape from non-optimal Pareto stationary points; in this case, NSGA-II could exploit its genetic operators to obtain better solutions.

Overall, NSGA-II outperformed FPG on the Γ–*spread* and Δ–*spread* metrics. Note that the Δ–*spread* metric is not available for FPG on the MAN_1 problem with $n = 5$, since it requires at least two points to be returned. However, as the value for n increases, the limited scalability of the genetic approach may worsen its performance. Some examples of this behavior are the *spread* metrics results of the two approaches on high dimensional instances of the CEC09_1 and CEC09_4 problems. In these cases, FPG obtained better performance: the constrained steepest partial descent directions and the B-FALS algorithm turned out to be helpful in exploring the extreme regions of the objectives space and, then, in finding a spread approximation of the Pareto front.

In conclusion, both algorithms have features that make them very effective in specific situations: on convex and not extremely difficult non-convex problems, FPG was better in obtaining good solutions, while NSGA-II was more effective on escaping from non-optimal points on highly irregular problems. Furthermore, the genetic features of NSGA-II let this latter one perform better in finding spread and uniform Pareto fronts most of the times: this fact is indeed reflected in the *spread* metrics values obtained by NSGA-II. However, the limited scalability of NSGA-II could compromise its performance on high dimensional problems, in terms of both *Purity* and *spread* metrics. All these facts remark once again how much trying to join these benefits in one algorithm might be appealing.

9.2.2 Preliminary Comparisons between NSMA and the State-of-the-art Algorithms

In this section, we provide the results on two problems along with some first comments about the behavior of the four algorithms. We analyzed the CEC09_3 problem with $n = 10$ and the ZDT_3 problem with $n = 20$. The first one has particularly difficult objective functions, while the second one is also characterized by a composite function and a disconnected front which is not convex everywhere. We consider these problems suitable to start an analysis about the performance of the considered algorithms. For other information on CEC09_3 and ZDT_3, the reader is referred to Table 9.1.

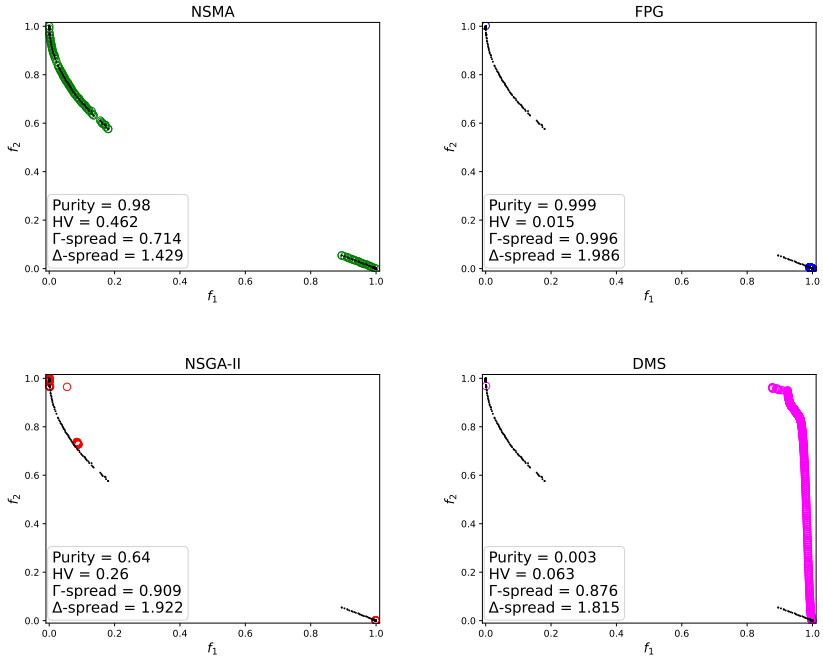


Figure 9.3: Approximation of the Pareto front of the CEC09_3 problem with $n = 10$.

From the results on the CEC09_3 problem, shown in Figure 9.3, we immediately observe the effectiveness of our approach. Indeed, NSMA outperformed

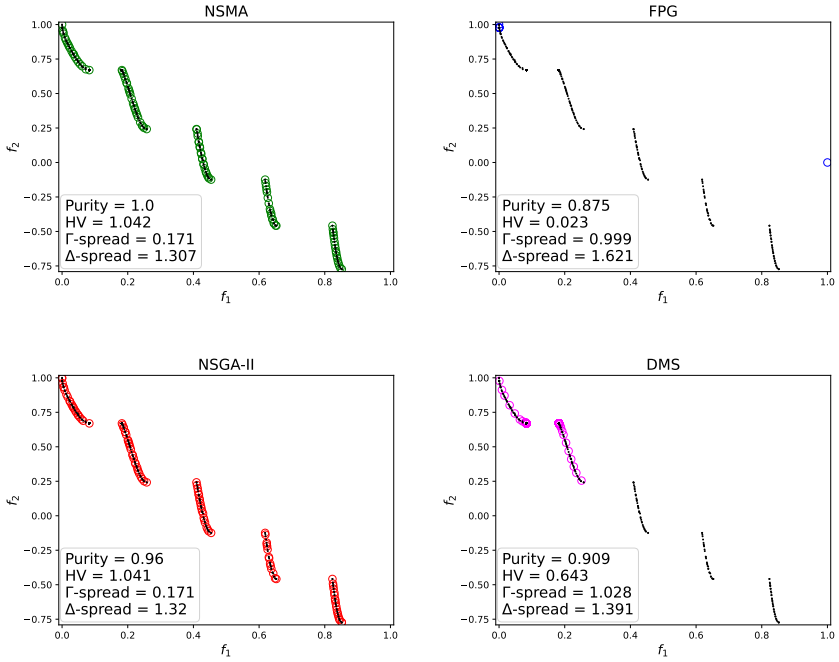


Figure 9.4: Approximation of the Pareto front of the ZDT_3 problem with $n = 20$.

the other algorithms in terms of *Hypevolume*, Γ*–spread* and Δ*–spread*. Moreover, its result on the *Purity* metric was similar to the best one, which was obtained by `FPG`, and better than the two other competitors.

`NSGA-II` and `FPG` turned out to be the second and the third best algorithms, respectively, with `FPG` outperforming the genetic method only in terms of *Purity*. The two algorithms seem not to be capable of spreading the search in the objectives space. Indeed, they retrieved many points but most of them are concentrated in a small portion of the objectives space. In this regard, `NSMA` was better: this result arguably comes from the use of constrained steepest partial descent directions with points characterized by a high crowding distance. Indeed, using descent steps at such points lets `NSMA` obtain a more spread and uniform Pareto front approximation w.r.t. its competitors.

NSMA and NSGA-II turned out to be the best algorithms on the ZDT_3 problem, as we can observe in Figure 9.4. Furthermore, they exhibited very similar performance. It is known that NSGA-II is one of the most effective algorithms to use with the ZDT problem class. Indeed, its genetic features allow it to escape from non-optimal Pareto-stationary solutions and to obtain good results with the most complex functions. NSMA seems to use these features as efficiently as NSGA-II. We also observe a little performance enhancement in terms of *Purity*.

The lack of these characteristics did not allow FPG to have the same performance. Indeed, although this algorithm obtained a good value for the *Purity* metric, it produced few points and it was not capable to obtain a spread and uniform Pareto front. DMS seems not to have the same issues, having been able to properly identify two blocks of the disconnected front. In this case, the derivative-free algorithm even managed to obtain a slightly better *Purity* value than FPG.

9.2.3 Performance Analysis in Variable Settings

In this section, we want to assess the robustness of the proposed algorithm in the specific settings where, as highlighted in Section 9.2.1, genetic and descent methods exhibit particular struggles. In detail, we compare the performance of the four algorithms (NSMA, FPG, NSGA-II and DMS) in two peculiar problems already addressed in Section 9.2.1: MAN_1 (F convex) and CEC09_4 (F nonconvex). Moreover, we consider the following problem dimensionalities: $n \in \{5, 10, 20, 200\}$.

The results for the MAN_1 problem are shown in Figure 9.5 and Table 9.5. For $n = 5$, DMS turned out to be the best overall algorithm, obtaining a *Purity* value similar to the one of FPG and excelling in terms of *Hypervolume* and Γ–*spread*. However, observing the plot, the points produced by the NSMA algorithm seem to be near to those obtained by DMS and FPG. We hence deduce that the latter algorithms produced only slightly better points. Furthermore, in this problem NSMA outperformed the competitors in terms of Δ–*spread*: our method managed to achieve a more uniform Pareto front.

As the value of n increases, the DMS performance gets worse and NSMA outperforms it w.r.t. all the metrics. In particular, in these cases our method turned out to be the best in terms of the *Hypervolume* and *spread* metrics. For $n \in \{10, 20\}$, DMS produced only a single point that is also dominated (it is not observable in the figure since it is too far from the reference front).

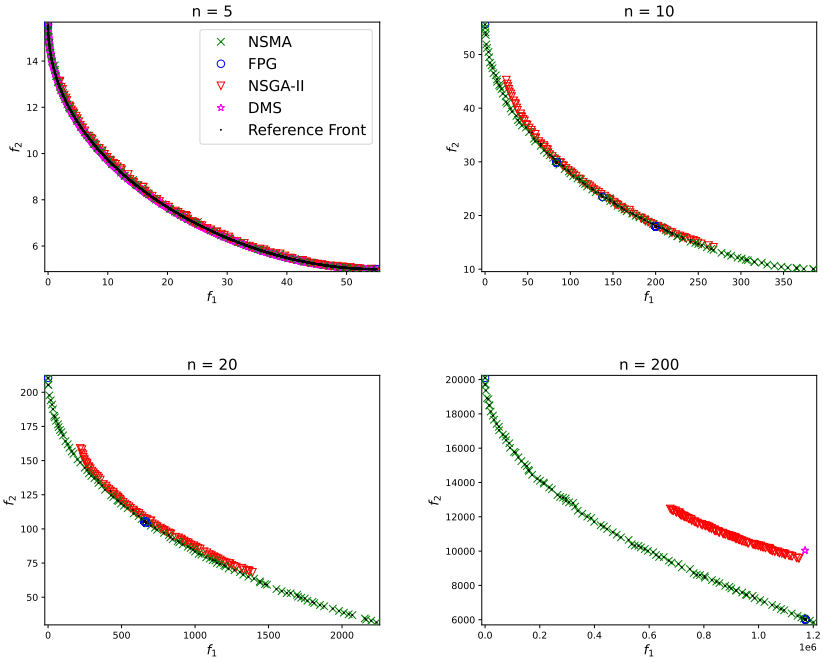


Figure 9.5: Approximation of the Pareto front of the convex MAN_1 problem at different dimensionalities retrieved by `NSMA`, `FPG`, `NSGA-II` and `DMS`.

In fact, the performance drop of `DMS` as the size of problems grows is not unexpected: derivative-free algorithms based on searches along coordinate directions are well known to poorly scale in general.

The performance of `NSGA-II` is rather poor w.r.t. `NSMA`, regardless the value of n. Arguably, this result can be attributed to the aforementioned `NSGA-II` performance slowdown occurring on problems characterized by a particularly large feasible sets (Section 4.2.1.1). In this context, `NSMA` particularly exploited the surrogate bounds, the constrained steepest descent directions and the optimization of the points with high crowding distance. The constrained steepest descent directions also allowed `FPG` to be the best algorithm in terms of *Purity* overall. However, this method poorly performed regarding the *Hypervolume* and *spread* metrics. Finally, for great values of n, our approach and `FPG` are the only algorithms whose *Purity* values are

n	Metric	NSMA	FPG	NSGA-II	DMS
5	*Purity*	0.38	**1.0**	0.01	0.995
	Hypervolume	444.767	0.0	440.096	**448.565**
	Γ*–spread*	2.231	55.0	2.439	**0.027**
	Δ*–spread*	**0.624**	N/A	0.655	0.902
10	*Purity*	0.94	**1.0**	0.45	0.0
	Hypervolume	**4.637 x 10^{10}**	4.632 x 10^{10}	4.635 x 10^{10}	0.0
	Γ*–spread*	**11.212**	185.201	117.139	5.968 x 10^{6}
	Δ*–spread*	**0.566**	1.803	0.661	N/A
20	*Purity*	0.99	**1.0**	0.02	0.0
	Hypervolume	**3.746 x 10^{10}**	3.716 x 10^{10}	3.731 x 10^{10}	0.0
	Γ*–spread*	**84.651**	1568.5	843.579	4.071 x 10^{6}
	Δ*–spread*	**0.48**	1.584	0.7	N/A
200	*Purity*	0.99	**1.0**	0.0	0.0
	Hypervolume	**1.135 x 10^{10}**	4.127 x 10^{8}	4.898 x 10^{9}	3.11 x 10^{8}
	Γ*–spread*	**29545.758**	1171289.94	677249.587	1.17 x 10^{6}
	Δ*–spread*	**0.492**	1.948	0.9	1.0

Table 9.5: Metrics values achieved by the four algorithms (NSMA, FPG, NSGA-II and DMS) on the convex MAN_1 problem for $n \in \{5, 10, 20, 200\}$. The metric values marked in bold are the best obtained on the considered instance.

not near to 0.

Regarding the CEC09_4 problem, whose results are reported in Figure 9.6 and Table 9.6, NSMA was the algorithm with the best overall performance: it generally obtained better metrics values than its most important competitors (FPG and NSGA-II). Here, the combination of genetic operations and constrained steepest descent directions was greatly helpful to escape from non-optimal Pareto-stationary points and, thus, to obtain remarkable results. Indeed, the independent use of only one of these two approaches did not lead to the same performance. In this problem, except in terms of *Purity* in the $n = 10$ case, the DMS algorithm performed poorly regardless the considered dimensionality.

In conclusion, NSMA can be considered a viable option with convex problems, both in the low and the high dimensional cases. At the same time, our approach did not suffer with non-convex problems characterized by difficult objective functions, as opposed to FPG. On the contrary, it also outperformed

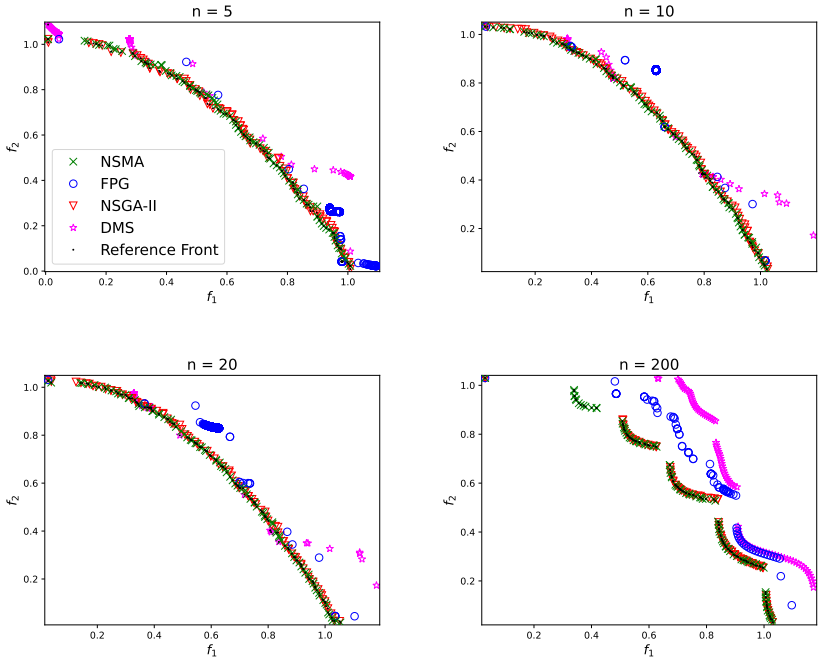


Figure 9.6: Approximation of the Pareto front of the nonconvex CEC09_4 problem at different dimensionalities retrieved by `NSMA`, `FPG`, `NSGA-II` and `DMS`.

`NSGA-II`, which is known to be a particularly suitable algorithm to use in these cases but struggles as the dimensionality of the problem grows.

9.2.4 Overall Comparison

In this last section of computational experiments, we provide the performance profiles for the four considered algorithms on the CEC09, ZDT, MOP_1, MOP_2, MOP_3 and MAN_1 problems. The reader can be found them listed in Table 9.1. The profiles are shown in Figure 9.7.

The performance profiles remark once again the benefits of using our proposed approach. Regarding the *Purity* metric: `NSMA` is the clear winner. In problems with complicated objective functions, local optimization of points

n	Metric	NSMA	FPG	NSGA-II	DMS
5	*Purity*	**0.65**	0.025	0.58	0.0
	Hypervolume	0.473	0.365	**0.477**	0.375
	Γ–*spread*	**0.121**	0.421	0.134	0.331
	Δ–*spread*	**0.582**	1.824	0.789	1.946
10	*Purity*	0.78	0.002	0.55	**0.883**
	Hypervolume	**0.5**	0.404	0.496	0.384
	Γ–*spread*	0.05	0.3	**0.03**	0.289
	Δ–*spread*	**0.539**	1.973	0.551	1.734
20	*Purity*	**0.82**	0.0	0.6	0.112
	Hypervolume	**0.487**	0.391	0.483	0.382
	Γ–*spread*	0.103	0.339	**0.087**	0.301
	Δ–*spread*	**0.539**	1.917	0.576	1.644
200	*Purity*	**0.8**	0.011	0.75	0.0
	Hypervolume	**0.427**	0.306	0.407	0.234
	Γ–*spread*	**0.31**	0.451	0.478	0.599
	Δ–*spread*	**0.992**	1.208	1.08	1.111

Table 9.6: Metrics values achieved by the four algorithms (NSMA, FPG, NSGA-II and DMS) on the nonconvex CEC09_4 problem for $n \in \{5, 10, 20, 200\}$. The metric values marked in bold are the best obtained on the considered instance.

in the NSMA mechanisms could result in a waste of computational time. From the results, however, we deduce that the converse is true: the combined use of constrained steepest descent directions and genetic operations allowed NSMA to achieve the best performance.

The proposed method also outperformed the other ones in terms of *Hypervolume* and Γ–*spread*, while its performance is very similar to the one of NSGA-II in terms of Δ–*spread*. We can conclude that our approach is able to effectively obtain accurate, spread and uniform Pareto front approximations. At the same time, we deduce that the same cannot be said for the FPG, which turned out to be the worst method w.r.t. these metrics. However, the descent-based algorithm was the second best in terms of *Purity*, outperforming NSGA-II. In general, DMS was not effective overall on the considered benchmark.

Lastly, we tested the four algorithms considering a time limit of 30 seconds for the experiments: the results can be seen in Figure 9.8. Our aim is

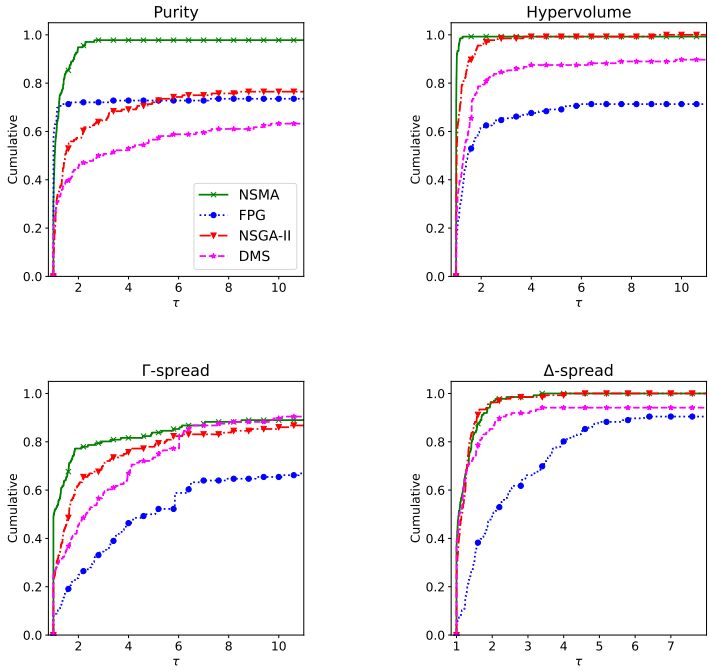


Figure 9.7: Performance profiles for the `NSMA, FPG, NSGA-II` and `DMS` algorithms on the CEC09, ZDT, MOP_1, MOP_2, MOP_3 and MAN_1 problems, run with a time limit of 2 minutes.

to observe the effectiveness of the methods at the first iterations.

Considering the *Hypervolume* and the Γ*–spread* metrics, we observe that the differences between our approach and the other algorithms are now even clearer, while the situation is not changed in terms of *Purity* and Δ*–spread*. We can conclude that `NSMA` turned out to be also effective considering a smaller time limit: from the very first iterations, our approach was capable to obtain good, wide and uniform Pareto front approximations.

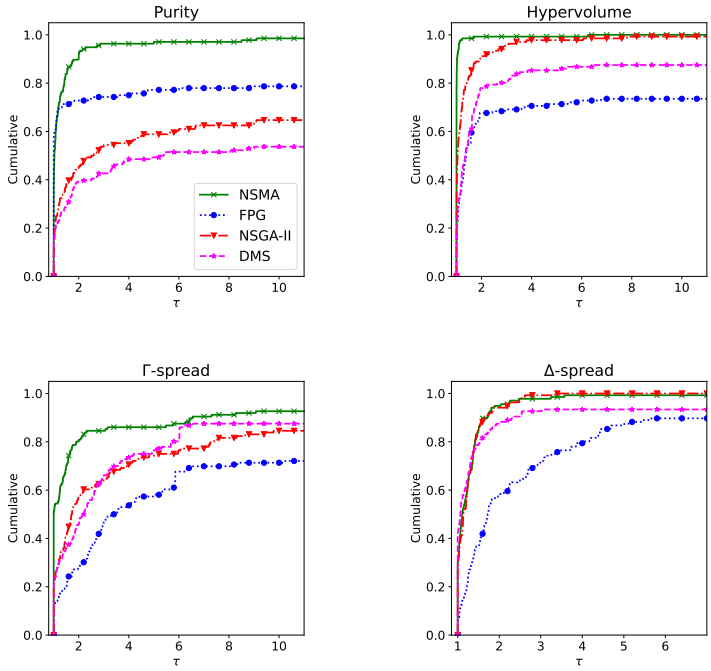


Figure 9.8: Performance profiles for the `NSMA, FPG, NSGA-II` and `DMS` algorithms on the CEC09, ZDT, MOP_1, MOP_2, MOP_3 and MAN_1 problems, run with a time limit of 30 seconds.

9.3 Computational Experiments on the Limited Memory Quasi-Newton Approach for MOO

In this section [2], we compare the performance of the limited memory Quasi-Newton approach proposed in Section 5.1 (Chapter 5), which we call `LM-Q-NWT`, and of other state-of-the-art Newton and Quasi-Newton methods for MOO from the literature.

In particular, the first competitor is the Multi-Objective Newton method

[2]The full code of the experiments can be found at `https://github.com/pierlumanzu/limited_memory_method_for_MOO` [76].

(`NWT`) proposed in [34]. Since this method is not designed to handle unconstrained multi-objective non-convex problems, we evaluated its performance only on the convex test instances. The other two competitors are the Quasi-Newton approach (`Q-NWT`) proposed in [90] and the Modified Quasi-Newton method (`MQ-NWT`) presented in [1]. We refer the reader to Section 3.1.3 for additional details on these three approaches. At the first iteration of all the Quasi-Newton approaches, including `LM-Q-NWT`, the approximation matrix/-matrices is/are set equal to the identity matrix.

In Table 9.7, we list the problems tested in the experiments of this section. In particular, we compared the algorithms in 78 convex and 83 non-convex problems. Additional details on these latter ones can be also found in Table 9.1. The problems scalable w.r.t. the problem dimension were tested for values of $n \in \{2, 5, 10, 20, 30, 40, 50, 100, 200, 500, 1000\}$, except for the CEC09 problems where the $n = 2$ case was not considered. For each algorithm, we tested each problem with 100 different initial points chosen from a uniform distribution. The latter was defined through lower and upper bounds specified for each problem in Table 9.7. Since in this context we consider *unconstrained* multi-objective optimization problems, these bounds were only used to choose the random initial points. Starting from an initial point, we decided to let the algorithms run until one of the following stopping conditions was met:

- the current solution is ε-Pareto-stationary (Definition 2.4); in the experiments,
$$\varepsilon = 5\texttt{eps}^{1/2},$$
where `eps` denotes the machine precision;
- a time limit of 2 minutes is reached.

In order to make the comparisons as fair as possible, we decided to use the same line search strategy for all the approaches. In particular, we employed the Wolfe line search proposed in Section 5.1.3. Note that all the considered problems have objective functions that let Assumption 3.1 with $\Omega = \mathbb{R}^n$ hold and, thus, the finite termination of Algorithm 5.3 is guaranteed. The values for the line search parameters were chosen according to some experiments on a subset of the tested problems and are as follows: $\gamma = 10^{-4}$, $\sigma = 10^{-1}$, $\eta = 2.5$ and $\delta = 0.5$. We do not report these preliminary results for the sake of brevity. In order to efficiently use the proposed line search in `MQ-NWT`, we used Equation (5.7) to compute ρ^k at each iteration k.

Type	Problem	Bounds
Convex	JOS_1a	$[-10, 10]^n$
	JOS_1b	$[-10^2, 10^2]^n$
	JOS_1c	$[10^{-2}, 1]^n$
	SLC_2	$[-10^2, 10^2]^n$
	M-FDS_1	see App. D
	MOP_7	see [50]
	MAN_1	$[-10, 10]^n$
	MAN_2	see App. D
Non Convex	CEC09_1, CEC09_2, CEC09_3 CEC09_7, CEC09_8, CEC09_10	see [107]
	M-MOP_2	see App. D
	MOP_3	see [50]
	MMR_5	see [80]

Table 9.7: Problems tested in the computational experiments of Section 9.3, along with the bounds used to choose the initial points.

The choice for the parameter M of the new limited memory approach is separately discussed in Section 9.3.1. Since it denotes the number of vectors pairs maintained in memory during the iterations, it is the most critical among the `LM-Q-NWT` parameters.

For each algorithm and problem, the main considered metrics are the following.

- N_ε: the percentage of runs ended with an ε-Pareto-stationary point.
- T: the computational time to reach the ε-Pareto-stationarity from an initial point. If the ε-Pareto-stationarity is not reached within the time limit, the value of T related to that point is set to ∞.
- T_M: the mean of the finite T values.

Similar to *Purity* and *Hypervolume* (Section 9.1.1), N_ε has increasing values for better solutions. Thus, the performance profiles w.r.t. this metric were produced based on the inverse of the obtained values.

9.3.1 Selection of the Parameter M

The parameter M indicates how many vectors pairs $\{(s_i, u_i)\}$ are maintained in memory at each iteration of `LM-Q-NWT`. A bad value for this parameter

might compromise the overall performance of the approach, making it too slow or not capable of reaching ε-Pareto-stationary points within the time limit.

In order to select a proper value for M, we analyzed the performance of `LM-Q-NWT` with $M \in \{2, 3, 5, 10, 20\}$ on a subset of the tested problems.

- 2 convex problems: SLC_2 ($m = 2$), MAN_2 ($m = 3$).
- 2 non-convex problems: CEC09_1 ($m = 2$), CEC09_10 ($m = 3$).

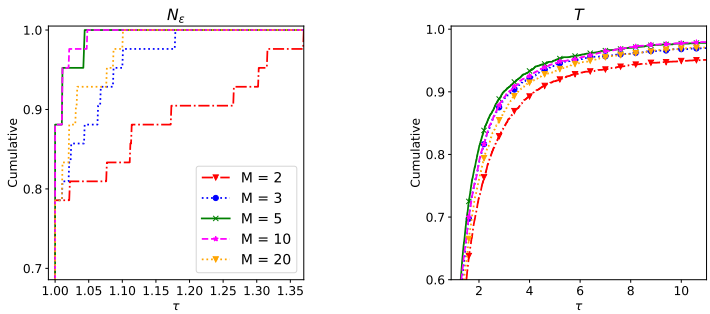


Figure 9.9: Performance profiles for the `LM-Q-NWT` algorithm with $M \in \{2, 3, 5, 10, 20\}$ on the SLC_2, MAN_2, CEC09_1 and CEC09_10 problems.

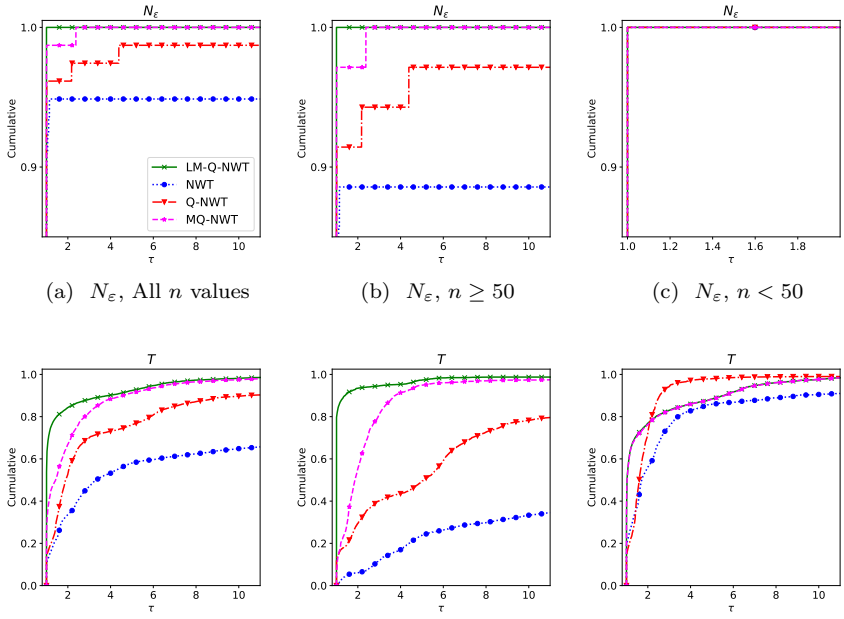
In Figure 9.9, we report the performance profiles for the five variants of the new limited memory method. The solvers with $M \in \{5, 10\}$ turned out to be the best w.r.t. both N_ε and T, while the variant with $M = 2$ was outperformed by all the other methods. We conclude that too little information on the previous steps can compromise the performance of `LM-Q-NWT`. On the other hand, the management of too many vectors pairs and the use of the two-loop recursive procedure can require great computational costs. A demonstration of this fact is the performance of the proposed approach with $M = 20$ on the T metric. Although this solver performed well w.r.t. N_ε, it is only the fourth most robust algorithm in terms of computational time.

After analyzing the performance profiles, we decided to use the new limited memory approach with $M = 5$ for the rest of the section. However, the variant with $M = 10$ appears to be a good choice too.

9.3.2 Overall Comparisons

In this section, we compare the proposed approach with the Newton and Quasi-Newton algorithms. As already mentioned, we tested `NWT` only on the convex problems. Then, we separately report the performance profiles for the convex and non-convex problems in Figures 9.10 and 9.11 respectively. In order to better remark the differences among the methods, for each metric we show three plots concerning different sets of values for n.

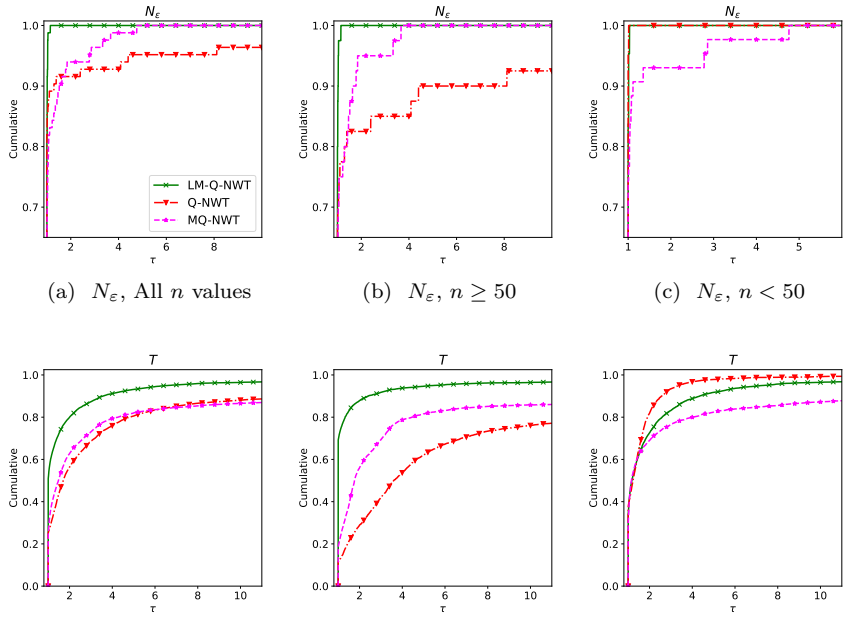
- Figures 9.10a, 9.10d, 9.11a, 9.11d: all the n values.
- Figures 9.10b, 9.10e, 9.11b, 9.11e: $n \geq 50$.
- Figures 9.10c, 9.10f, 9.11c, 9.11f: $n < 50$.



(a) N_ε, All n values (b) N_ε, $n \geq 50$ (c) N_ε, $n < 50$

(d) T, All n values (e) T, $n \geq 50$ (f) T, $n < 50$

Figure 9.10: Performance profiles for the `LM-Q-NWT`, `NWT`, `Q-NWT` and `MQ-NWT` algorithms on the convex problems of Table 9.7.



(a) N_ε, All n values (b) N_ε, $n \geq 50$ (c) N_ε, $n < 50$

(d) T, All n values (e) T, $n \geq 50$ (f) T, $n < 50$

Figure 9.11: Performance profiles for the `LM-Q-NWT`, `Q-NWT` and `MQ-NWT` algorithms on the non-convex problems of Table 9.7.

Regarding the performance on the convex problems for all the n values, the proposed approach proved to be the best algorithm, outperforming the competitors w.r.t. both the metrics. Moreover, the gap between `LM-Q-NWT` and the others is sharper when taking into account the non-convex problems or the high dimensional ones. For high n values, the `NWT` and `Q-NWT` algorithms proved to suffer the maintenance of the Hessians and the approximation matrices respectively. As a consequence, they turned out to be the least robust w.r.t. both the metrics. Using a single approximation matrix allowed the `MQ-NWT` approach to perform better. However, in extremely high dimensional problems, even managing a single matrix proved to be an expensive job. In these cases, the performance of the limited memory approach was remarkable.

On the low dimensional problems, the `NWT` and `Q-NWT` algorithms had a

good performance. The proposed approach similarly behaved w.r.t. the N_ε metric, but it was generally outperformed by these algorithms in terms of T. Managing the real Hessians or the approximation matrices turned out to be a tractable task when n is small enough. Moreover, by definition, these matrices provide more accurate information about the curvature of the objective functions than the matrix of `LM-Q-NWT` and `MQ-NWT`. However, these two algorithms still proved to be competitive, obtaining good T metric results in most of the problems.

In order to analyze the performance of the algorithms more deeply, in Tables 9.8-9.9-9.10-9.11 we report the metrics values obtained in two convex and two non-convex problems respectively. In particular, we show the results for $n \in \{5, 20, 50, 200, 500, 1000\}$.

n	$\mathbf{N_\varepsilon}$				$\mathbf{T_M}$			
	LM-Q-NWT	NWT	Q-NWT	MQ-NWT	LM-Q-NWT	NWT	Q-NWT	MQ-NWT
5	**1.0**	**1.0**	**1.0**	**1.0**	**0.028**	0.102	0.045	**0.028**
20	**1.0**	**1.0**	**1.0**	**1.0**	**0.015**	0.586	0.093	**0.015**
50	**1.0**	**1.0**	**1.0**	**1.0**	**0.022**	1.113	0.218	**0.022**
200	**1.0**	**1.0**	**1.0**	**1.0**	**0.081**	14.604	0.696	0.088
500	**1.0**	0.88	**1.0**	**1.0**	**0.187**	90.595	1.041	0.229
1000	**0.57**	0.01	0.13	0.24	52.184	146.784	**28.078**	68.779

Table 9.8: Metrics values achieved by the `LM-Q-NWT`, `NWT`, `Q-NWT` and `MQ-NWT` algorithms on the convex MAN_1 problem ($m = 2$) for $n \in \{5, 20, 50, 200, 500, 1000\}$. A value marked in bold is the best obtained for a metric on a specific problem.

Regarding the N_ε metric, the proposed method outperformed the competitors regardless the values for n and m. As in the performance profiles, the differences between `LM-Q-NWT` and the other approaches are clearer on the high dimensional problems. In some of these, `NWT` and `Q-NWT` were not able to obtain any ε-Pareto-stationary point.

On the problems with two objective functions, almost all the best results in terms of the T_M metric were obtained by the proposed approach. However, the same performance was not obtained on the problems with $m = 3$ and low value for n. The use of a single matrix seems not to provide accurate enough information about the functions curvature when the objectives are more than two. An additional demonstration of this fact could be also the similar performance of the `MQ-NWT` algorithm. On the other hand, the use of

n	$\mathbf{N}_\varepsilon$				$\mathbf{T}_\mathbf{M}$			
	LM-Q-NWT	NWT	Q-NWT	MQ-NWT	LM-Q-NWT	NWT	Q-NWT	MQ-NWT
5	**1.0**	**1.0**	**1.0**	**1.0**	0.771	**0.132**	0.176	0.758
20	**1.0**	**1.0**	**1.0**	**1.0**	2.413	**0.32**	0.403	2.44
50	**1.0**	**1.0**	**1.0**	**1.0**	3.376	0.891	**0.688**	3.287
200	**1.0**	**1.0**	**1.0**	**1.0**	5.333	13.416	**5.292**	13.113
500	**1.0**	0.95	**1.0**	**1.0**	**23.353**	109.853	36.66	39.028
1000	**1.0**	0.0	0.0	0.99	**32.868**	-	-	109.541

Table 9.9: Metrics values achieved by the LM-Q-NWT, NWT, Q-NWT and MQ-NWT algorithms on the convex M-FDS_1 problem ($m = 3$) for $n \in \{5, 20, 50, 200, 500, 1000\}$. A value marked in bold is the best obtained for a metric on a specific problem.

n	$\mathbf{N}_\varepsilon$			$\mathbf{T}_\mathbf{M}$		
	LM-Q-NWT	Q-NWT	MQ-NWT	LM-Q-NWT	Q-NWT	MQ-NWT
5	**1.0**	**1.0**	**1.0**	**0.057**	0.093	**0.057**
20	**1.0**	**1.0**	**1.0**	0.116	0.166	**0.113**
50	**1.0**	**1.0**	**1.0**	**0.142**	0.225	0.145
200	**1.0**	**1.0**	**1.0**	**0.324**	1.158	0.495
500	**1.0**	**1.0**	**1.0**	**1.013**	5.766	1.673
1000	**1.0**	**1.0**	**1.0**	**1.787**	32.464	5.244

Table 9.10: Metrics values achieved by the LM-Q-NWT, Q-NWT and MQ-NWT algorithms on the non-convex M-MOP_2 problem ($m = 2$) for $n \in \{5, 20, 50, 200, 500, 1000\}$. A value marked in bold is the best obtained for a metric on a specific problem.

the real Hessian/an approximation matrix for each objective function seems to overcome the issue: indeed, NWT and Q-NWT had the best performance in terms of T_M in these cases. LM-Q-NWT still obtained great results for this metric on the problems with three objective functions and high value for n, outperforming the other competitors. Even with $m = 3$, the employment of a single matrix turned out to be essential in high dimensional problems. Like the proposed approach, MQ-NWT proved to perform better than NWT and Q-NWT with $m = 3$ and high value for n, resulting the second best algorithm in these cases.

n	$\mathbf{N}_\varepsilon$			$\mathbf{T}_\mathbf{M}$		
	LM-Q-NWT	Q-NWT	MQ-NWT	LM-Q-NWT	Q-NWT	MQ-NWT
5	**1.0**	**1.0**	**1.0**	1.036	**0.62**	1.472
20	**1.0**	**1.0**	0.99	3.4	**1.357**	3.72
50	**1.0**	**1.0**	**1.0**	5.531	**2.013**	6.213
200	**1.0**	**1.0**	**1.0**	**6.565**	12.028	11.798
500	**0.99**	0.9	0.98	**26.099**	80.713	36.693
1000	**0.94**	0.0	0.73	**32.367**	-	77.311

Table 9.11: Metrics values achieved by the LM-Q-NWT, Q-NWT and MQ-NWT algorithms on the non-convex CEC09_8 problem ($m = 3$) for $n \in \{5, 20, 50, 200, 500, 1000\}$. A value marked in bold is the best obtained for a metric on a specific problem.

9.3.3 Results in a Global Optimization Setting

In the previous section, we compared the LM-Q-NWT method with strongly related approaches from the state-of-the-art, in terms of efficiency and effectiveness at reaching approximate Pareto-stationarity. Now, we show the (positive) impact that the proposed procedure may have if used within a global multi-objective optimization framework. In particular, here we consider the memetic algorithm NSMA, proposed in Section 4.2 of this manuscript.

For the experiments, we consider two possible modifications of the NSMA algorithm:

- NSMA-W, which employs in the FMOPG method (Algorithm 4.6) the Wolfe line search proposed in Section 5.1.3;
- NSMA-L, which uses the new limited memory approach (Algorithm 5.1) as the local optimization procedure.

We compared these two approaches with NSGA-II (Section 4.1) and the original version of NSMA. In each variant of the memetic approach, the points selected as starting solutions for the local search procedures were only optimized w.r.t. all the objective functions. In the original version of NSMA, the points can be also refined w.r.t. a subset of the objective functions $\mathcal{I} \subset \{1, \ldots, m\}$. However, Assumption 3.1 ($\Omega = \mathbb{R}^n$) may not hold for some subset $\mathcal{I}$ and, then, when trying to optimize a point w.r.t. $\mathcal{I}$, the Wolfe line search would continue its execution for an infinite number of steps.

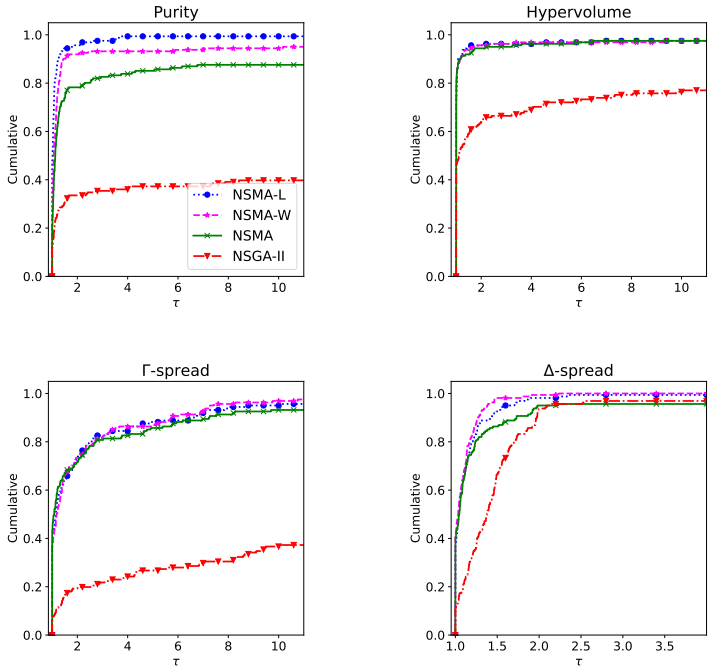


Figure 9.12: Performance profiles for the `NSMA-L`, `NSMA-W`, `NSMA` and `NSGA-II` algorithms on the problems of Table 9.7.

In Figure 9.12, we report the performance profiles for the `NSMA-L`, `NSMA-W`, `NSMA` and `NSGA-II` algorithms on the problems listed in Table 9.7. The considered performance metrics are *Purity*, *Hypervolume*, Γ*–spread* and Δ–spread. For more details on these metrics, the reader is referred to Section 9.1.1. We also remind that, due to the random operations contained both in `NSGA-II` and `NSMA`, all the algorithms were executed five times with different seeds for the pseudo-random number generator. Additional information on this experimental setting can be found in Section 9.1.2.

In terms of *Purity*, `NSMA-L` and `NSMA-W` turned out to be the two most robust algorithms. The proposed Wolfe line search allowed to improve the results of the original `NSMA`. In fact, the use of the limited memory approach allowed to obtain the best possible performance. Regarding the *Hypervolume* and *spread* metrics, `NSMA-L` and `NSMA-W` had a similar performance. `NSMA`

results on *Hypervolume* and Γ–*spread* are comparable with the ones of the two variants. However, the original approach was slightly outperformed in terms of Δ–*spread*. `NSGA-II` did not perform well w.r.t. all the metrics: the variants of `NSMA` turned out to be capable in finding more accurate and uniform Pareto front approximations.

9.4 Performance Assessment of the Improved Version of the Front Steepest Descent Algorithm

In this section [3], we show the results of computational experiments on the *Improved Front Steepest Descent* (`IFSD`) algorithm (Chapter 6), supporting the discussion in Sections 6.1-6.2.

In the experiments, we compared our approach (`IFSD`) to the original `FSD` (Algorithm 3.4), equipped with the base line search `FALS` (Algorithm 3.3) or the extrapolation strategy (`EFSD`). The parameters setting for the line searches can be found in Section 9.1.2.

The benchmark used for the comparisons consists of the unconstrained versions of the following problems: CEC09_2, CEC09_3, JOS_1b, MAN_1 ($m = 2$) and CEC09_10 ($m = 3$). The JOS_1b and MAN_1 problems are convex, whereas the CEC09 ones are nonconvex. Other information on them can be found in Table 9.1. For all the problems, we considered instances with values of n in $\{5, 10, 20, 30, 40, 50, 100, 200\}$. Moreover, each problem was tested twice, with different strategies for the initial points: a) n points are uniformly sampled from the hyper-diagonal defined by lower and upper bounds (see Section 9.1.3 for more details); b) only the midpoint of the hyper-diagonal is selected. We recall that the bounds of the JOS_1b problem can be found in Table 9.7.

We report in Figure 9.13 the performance profiles for the `IFSD`, `FSD` and `EFSD` algorithms on the entire benchmark of 80 problem instances. We observe a remarkable superiority of the proposed approach w.r.t. the original variants of the algorithm, especially in terms of the *spread* metrics, which points out that the Pareto front is indeed spanned more widely and uniformly. The strong *Hypervolume* performance also supports this result. As

[3] The implementation code of the `IFSD` algorithm can be found at `https://github.com/pierlumanzu/ifsd` [75]

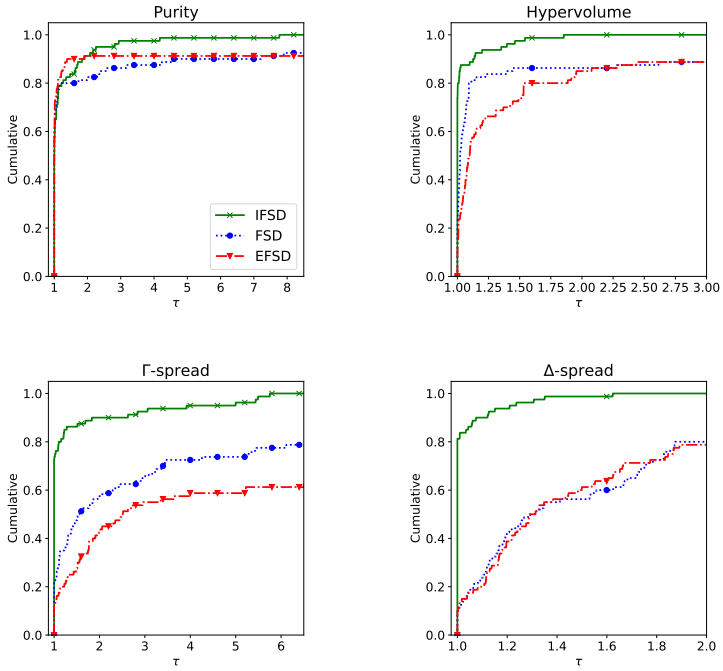


Figure 9.13: Performance profiles for the `IFSD`, `FSD` and `EFSD` algorithms on a benchmark of 80 MOO problems.

for *Purity* metric, the three algorithms appear to be closer, but we still observe a slight advantage of `IFSD`.

9.5 `FRONT-ALAMO` Performance Analysis

In this section [4], we focus on the comparisons between `FRONT-ALAMO` (Chapter 7) and some state-of-the-art methods in the multi-objective constrained optimization context, i.e., `MOSQP`, `NSGA-II` and `DMS`. These algorithms are already introduced in Section 9.1.2.

The problems considered for the experiments were mainly those with

[4] The implementation code of the `FRONT-ALAMO` algorithm can be found at `https://github.com/pierlumanzu/front-alamo` [77]

Problem(s)	Initial Point(s)
M-BNH_1, LAP_1, LAP_2	$\mathbf{0}_n$
M-BNH_2	$[8, -3]^\top$
M-OSY	$[2, 0, 1, 0, 1, 8]^\top$
CEC09, ZDT, MOP_1 MOP_2, MOP_3	See Sec. 9.1.3

Table 9.12: Initial points for the tested problems.

general convex constraints, i.e., the LAP, M-BNH and M-OSY problems. Then, we also tested the approaches on the bound-constrained CEC09, ZDT, MOP_1, MOP_2, MOP_3 problems. As in Section 9.1.3, it is worth remarking that the CEC09 and ZDT problems have particularly difficult objective functions, so they are interesting to study the effectiveness of the algorithms when solving hard problems. All these problems can be also found listed, along with other their characteristics, in Table 9.1. For each problem with general constraints, we started the algorithms from one feasible point (Table 9.12). In this way, we intended to study the exploration capabilities of the algorithms. Indeed, algorithms with great exploration abilities should create a spread and solid Pareto front on these problems. For the bound constrained problems, the initial points were uniformly selected from the hyper-diagonal, as already described in Section 9.1.3.

9.5.1 Preliminary Assessment of FRONT-ALAMO Performance w.r.t. ALAMO

As a preliminary computational experiment, we compared FRONT-ALAMO and ALAMO on a selection of the LAP_2 problems. As mentioned at the beginning of Chapter 7, FRONT-ALAMO is an extension of ALAMO [21] capable of dealing with sets of points and, then, of effectively producing Pareto front approximations of the considered problems. Thus, it is reasonable to compare the performance of our new approach w.r.t. its original version. Being ALAMO a single-point approach, it was run in a multi-start fashion from 100 initial randomly sampled feasible points. The execution was repeated 5 times, each of them with a different seed for the pseudo-random number generator, in order to be less sensitive to the random initialization. The resulting 5 fronts were compared based on the *Purity* metric and the best one was compared

with the front obtained by `FRONT-ALAMO`.

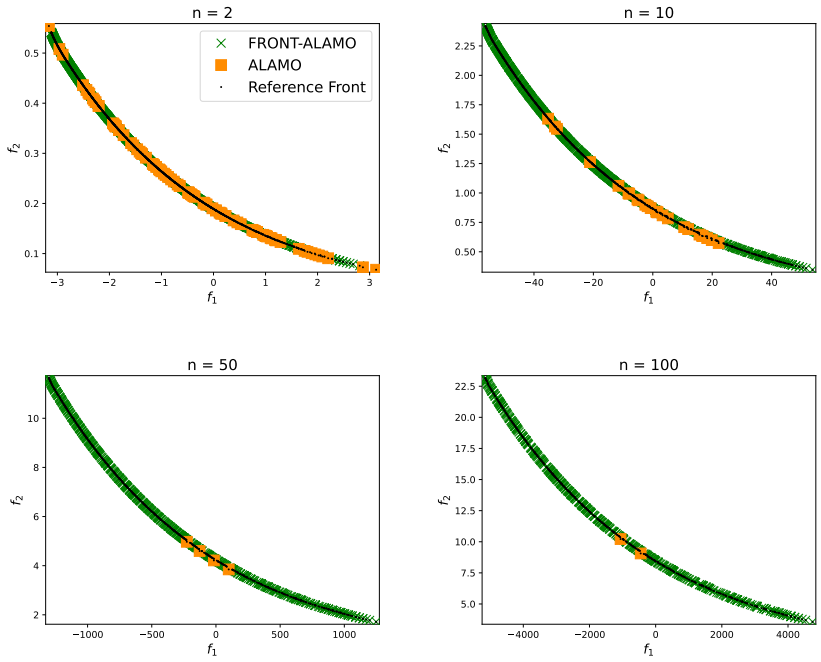


Figure 9.14: Pareto front approximation for `FRONT-ALAMO` and `ALAMO` considering LAP_2 problems at different dimensionalities.

In Figure 9.14 we show the plots of the fronts obtained by the two algorithms, while in Table 9.13 the related metrics values are reported. From the table, we can observe that `FRONT-ALAMO` was competitive in terms of *Purity* and outperformed `ALAMO` w.r.t. all the other metrics. The improvements on the *Hypervolume* and *spread* metrics are not surprising, since `FRONT-ALAMO` aims to approximate the Pareto front by iteratively exploiting every point of the current list, especially considering the current list itself to validate each line search step. The plots reflect the results in the table: as the value for n increased, `FRONT-ALAMO` managed to find more accurate, wider and more uniform Pareto front reconstructions than its single-point version.

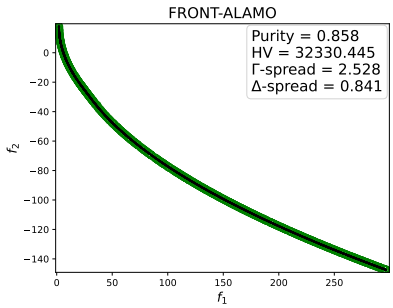
n	Metric	FRONT-ALAMO	ALAMO
2	*Purity*	0.959	**1.0**
	Hypervolume	**2.048**	2.011
	Γ–*spread*	**0.316**	0.669
	Δ–*spread*	**0.884**	0.905
10	*Purity*	0.907	**1.0**
	Hypervolume	**151.262**	135.687
	Γ–*spread*	**2.177**	31.607
	Δ–*spread*	**0.857**	0.949
50	*Purity*	0.962	**1.0**
	Hypervolume	**16709.004**	11244.539
	Γ–*spread*	**50.436**	1147.977
	Δ–*spread*	0.921	**0.889**
100	*Purity*	0.98	**1.0**
	Hypervolume	**127446.169**	81077.973
	Γ–*spread*	**208.059**	5194.112
	Δ–*spread*	**0.936**	0.941

Table 9.13: Metrics values obtained by FRONT-ALAMO and ALAMO in the LAP_2 problems with $n = 2, 10, 50, 100$. The values marked in bold are the best values (each of which is related to a specific score) obtained in a considered problem.

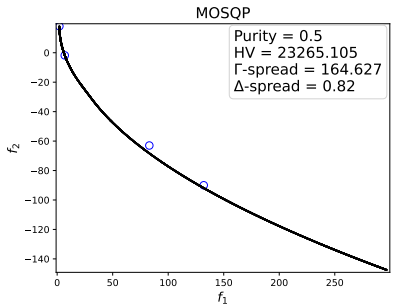
9.5.2 M-BNH, LAP_1 and M-OSY Problems

In this section, we study the performance of FRONT-ALAMO, MOSQP, NSGA-II and DMS on the M-BNH_1, M-BNH_2, LAP_1 and M-OSY problems.

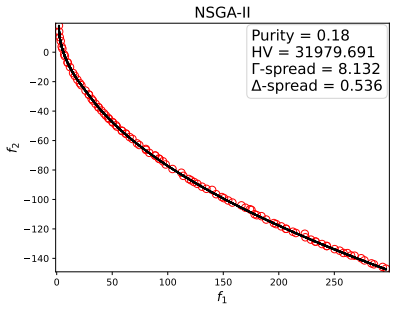
The results on the M-BNH problems (Figure 9.15) show the great performance of FRONT-ALAMO with respect to the competitors: indeed, our method obtained the second best *Purity* value both on the M-BNH_1 problem and on the M-BNH_2. In this latter problem, the differences with respect to our gradient-based competitor are even clearer, as MOSQP did not manage to obtain a single non-dominated solution w.r.t. the competitors. Considering the Δ–*spread*, FRONT-ALAMO was the second best method on the M-BNH_2 problem, while it was outperformed w.r.t this metric on the M-BNH_1 instance. However, in this last scenario, except for NSGA-II, the gap between the approaches is not too sharp. As for the *Hypervolume* and Γ–*spread* metrics, FRONT-ALAMO appears to have a decent behavior, being the second best algorithm on the M-BNH_1 problem and outperforming all the competitors on



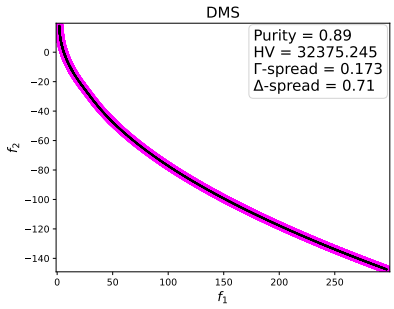
(a) `FRONT-ALAMO` Pareto front - M-BNH_1



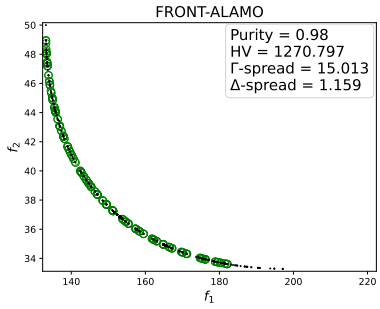
(b) `MOSQP` Pareto front - M-BNH_1



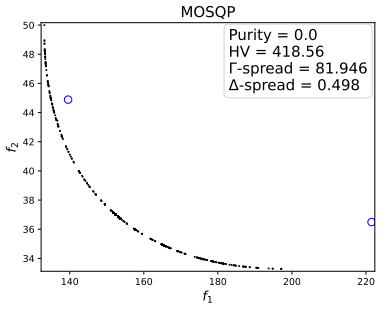
(c) `NSGA-II` Pareto front - M-BNH_1



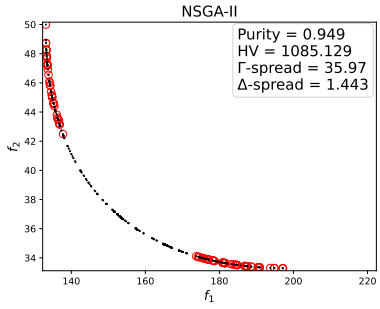
(d) `DMS` Pareto front - M-BNH_1



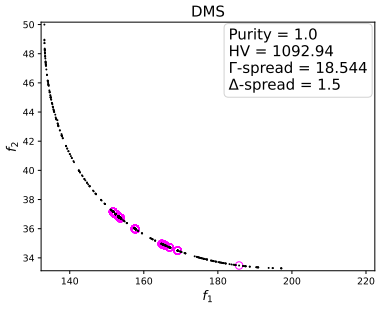
(e) `FRONT-ALAMO` Pareto front - M-BNH_2



(f) `MOSQP` Pareto front - M-BNH_2

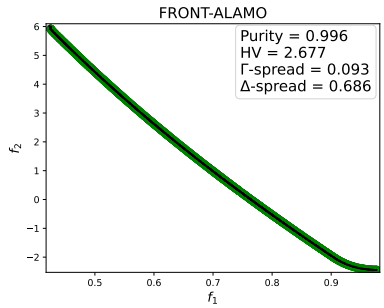


(g) `NSGA-II` Pareto front - M-BNH_2

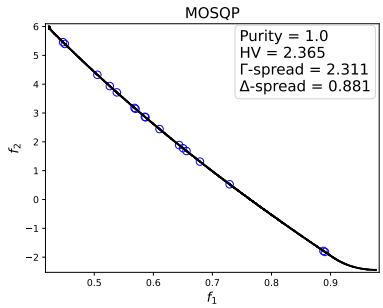


(h) `DMS` Pareto front - M-BNH_2

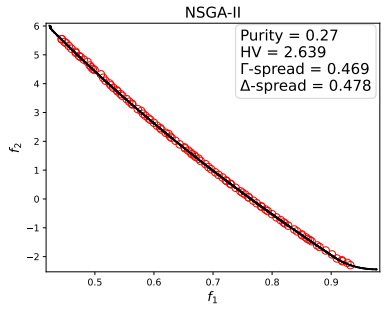
Figure 9.15: Pareto front approximation for the four algorithms considering the M-BNH problems.



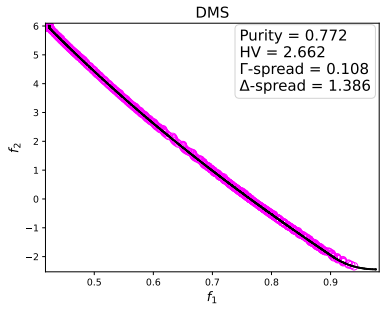
(a) FRONT-ALAMO Pareto front - LAP_1



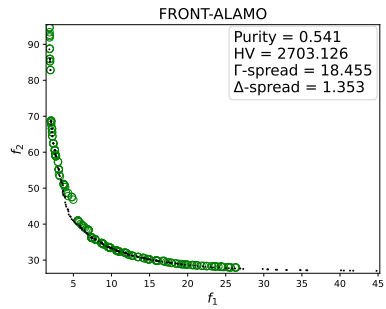
(b) MOSQP Pareto front - LAP_1



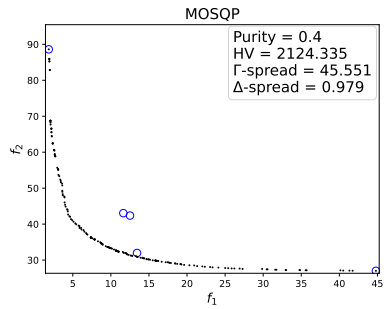
(c) NSGA-II Pareto front - LAP_1



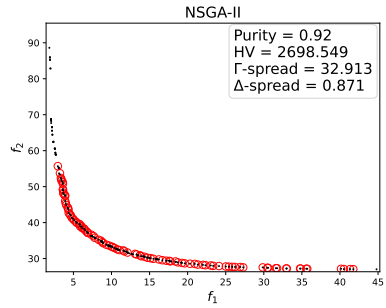
(d) DMS Pareto front - LAP_1



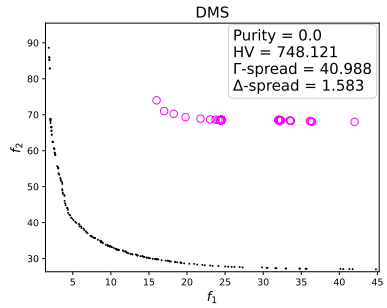
(e) FRONT-ALAMO Pareto front - M-OSY



(f) MOSQP Pareto front - M-OSY



(g) NSGA-II Pareto front - M-OSY



(h) DMS Pareto front - M-OSY

Figure 9.16: Pareto front approximation for the four algorithms considering the LAP_1 and M-OSY problems.

the M-BNH_2 problem. The worst algorithm turned out to be `MOSQP`, which seems to lack of search capabilities in the objectives space on the M-BNH problems. This fact can be also noted to a lesser extent for the `DMS` algorithm on the M-BNH_2 problem. On the M-BNH_1 instance, `DMS` outperformed the `NSGA-II` method in terms of *Purity*, *Hypervolume* and Γ–*spread*, while it is the opposite considering the Δ–*spread* metric. The situation is similar on the M-BNH_2 problem, although the performance of the two approaches was more similar.

Considering the LAP_1 problem (Figure 9.16a–d), `FRONT-ALAMO` performed very well, outperforming the competitors in terms of *Hypervolume* and Γ–*spread* and being the second best algorithm on the other metrics. The `MOSQP` method managed to outperform `FRONT-ALAMO` on one metric, that is the *Purity*, while the `NSGA-II` algorithm performed better on the Δ–*spread*, in terms of which the genetic algorithm turned out to be the best. `DMS` got results similar to the ones of our approach w.r.t. *Purity* and *Hypervolume*: this fact is also reflected in the front plots of the two approaches. In the M-OSY problem (Figure 9.16e–h) `NSGA-II` was the most effective obtaining an accurate and uniform Pareto front. In this case, `FRONT-ALAMO` achieved some interesting results. First of all, it outperformed the `DMS` algorithm w.r.t. all the metrics: this achievement is remarkable since `DMS` is gradient-free and can escape non optimal Pareto-stationary points, while our method is gradient-based. In addition, our algorithm obtained better values on *Purity*, *Hypervolume* and Γ–*spread* than its gradient-based competitor (`MOSQP`).

9.5.3 LAP_2 Problems

The LAP_2 problems represent another useful class of problems: indeed, they allow to discuss about the sensibility of the algorithms with regard to n, that is, how well they scale. Indeed, many algorithms have great performance considering small values of n. However, when a problem size grows, they lose their abilities to retrieve good Pareto front approximations.

Before seeing some plots and metric values, we show the performance profiles considering all the LAP_2 problems in Figure 9.17. The performance profiles highlight that `FRONT-ALAMO` outperformed the other competitors with respect to *Purity*, *Hypervolume* and Γ–*spread*. On the other hand, considering Δ–*spread*, `NSGA-II` was the best algorithm. In fact, the results of the methods on this metric differ little among each other: on the LAP_2 problems, no algorithm particularly suffered from a non-uniformity in their

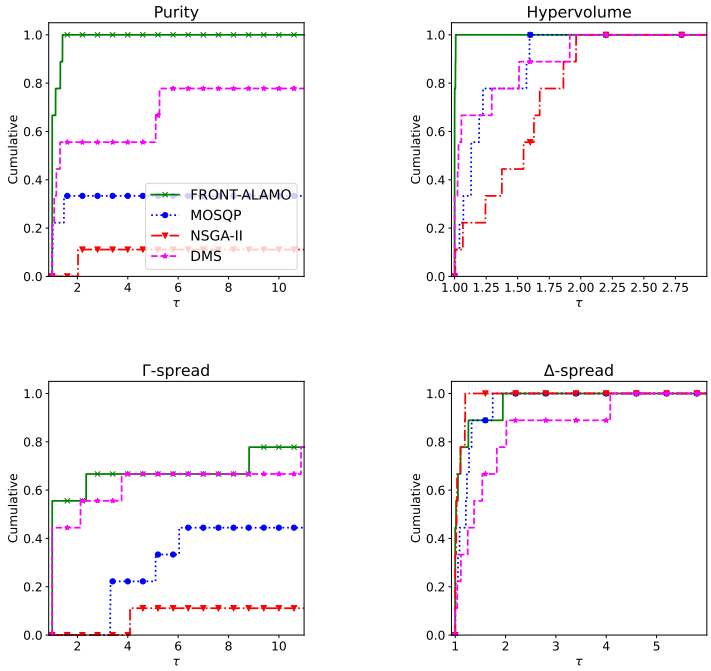


Figure 9.17: Performance profiles for the four algorithms on the LAP_2 problems.

fronts.

The motivation of these different results on the two *spread* metrics can be explained through Figure 9.18, where we show the Pareto fronts in four different LAP_2 problems. Here, we show the fronts all together in order to provide a more direct impression of the results. Indeed, in the LAP_2 problems, FRONT-ALAMO results show the superiority of our method at exploring the objectives space and creating a spread and uniform Pareto front. As the value for n increased, the competitors obtained large Γ–*spread* values with respect to those of FRONT-ALAMO, since they struggle to explore the extreme regions of the front.

When $n = 2$, all the methods managed to obtain the same Pareto front. However, increasing n, the differences between them become more and more clear. For instance, NSGA-II performance got worse with $n \geq 10$. The

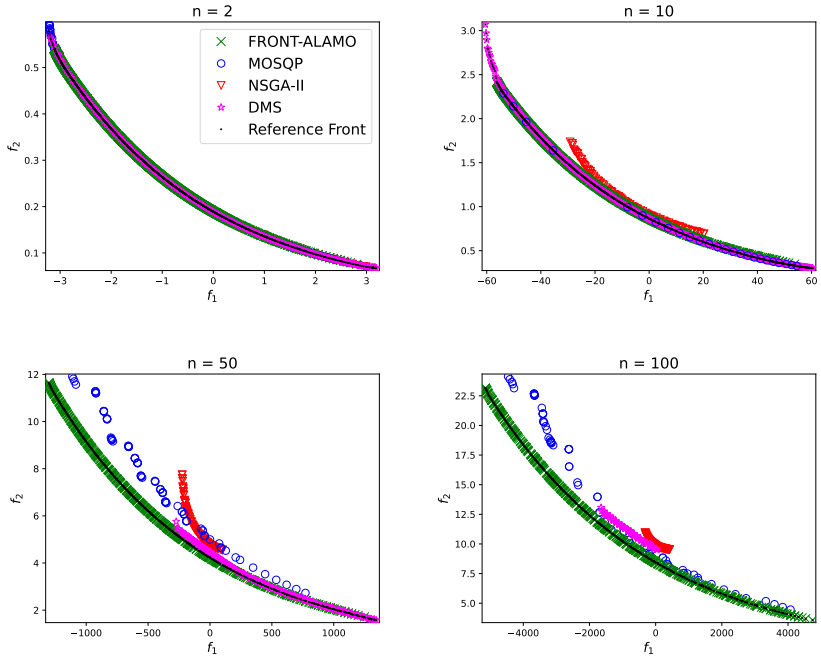


Figure 9.18: Pareto front approximation for the four algorithms considering LAP_2 problems at different dimensionalities.

genetic approach seems to be unable to spread the search in the objectives space and, also, to create a good, although small, Pareto front. The other gradient-free method (`DMS`) performed better but still it hardly reached the extremes of the objectives space when $n \geq 50$. `MOSQP` seems not to have this last negative feature but it generally retrieved very few points and, in addition, most of them are dominated. However, the `MOSQP` performance was good with $n \leq 10$.

The above comments on the algorithm behaviors on the LAP_2 problems are also supported by the numbers in Table 9.14. Observe that `FRONT-ALAMO` outperformed the competitors as the value of n increased. The superiority of our method when the dimension of a problem is high is very remarkable, especially considering *Purity*, *Hypervolume* and Γ*–spread*. As we just highlighted commenting the performance profiles, except for the $n = 2$ case, all

n	Metric	FRONT-ALAMO	MOSQP	NSGA-II	DMS
2	*Purity*	0.875	0.676	0.49	**0.991**
	Hypervolume	2.325	2.323	2.314	**2.326**
	Γ–*spread*	0.379	0.545	0.176	**0.043**
	Δ–*spread*	0.886	0.795	**0.456**	1.859
10	*Purity*	0.697	**0.92**	0.0	0.792
	Hypervolume	241.006	227.248	195.301	**243.002**
	Γ–*spread*	6.601	27.501	40.092	**0.427**
	Δ–*spread*	0.869	0.999	**0.825**	1.134
50	*Purity*	**0.995**	0.0	0.0	0.189
	Hypervolume	**18375.772**	15381.231	11285.47	14196.154
	Γ–*spread*	**94.525**	570.767	1240.803	1028.08
	Δ–*spread*	**0.923**	1.176	0.94	0.961
100	*Purity*	**1.0**	0.0	0.0	0.0
	Hypervolume	**137126.605**	121140.484	73609.626	90808.266
	Γ–*spread*	**208.059**	686.982	4827.8	4705.776
	Δ–*spread*	**0.936**	1.154	0.968	1.041

Table 9.14: Metrics values obtained by the four algorithms in the LAP_2 problems with $n = 2, 10, 50, 100$. The values marked in bold are the best values (each of which is related to a specific score) obtained in a considered problem.

the algorithms performed well regarding the Δ–*spread* metric.

9.5.4 CEC09, ZDT and MOP Problems

In this last section of computational experiments, we comment the results on the problems characterized only by boundary constraints listed at the beginning of Section 9.5. For the sake of brevity, we preferred to show the performance profiles related to all these problems in Figure 9.19.

The performance profiles on the *Purity* and *Hypervolume* metrics highlight the effectiveness of our method: it was not obvious, a priori, to obtain such great results with such complex functions, especially when some of our competitors are derivative-free and, thus, they potentially escape from non optimal Pareto-stationary points.

Considering the *spread* metrics, our results are competitive with respect to those of the other competitors. In particular, in the Γ–*spread* performance profiles FRONT-ALAMO was the third best algorithm, while the gradient-free

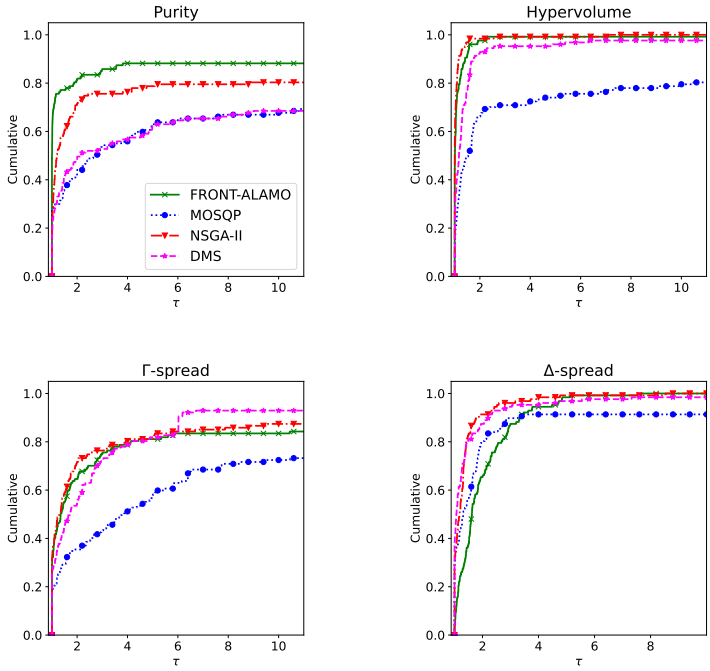


Figure 9.19: Performance profiles for the four algorithms on the CEC09, ZDT, MOP_1, MOP_2 and MOP_3 problems.

methods (`NSGA-II` and `DMS`) managed to have slightly better performance. Regarding the Δ–*spread*, `NSGA-II` and `DMS` turned out to be the most robust algorithms. However, the performance profiles on this metric are another proof of the effectiveness of the four algorithms to retrieve an uniform Pareto front.

9.6 Performance Evaluation of `MOIHT` and `SFSD`

In this section [5], we evaluate the efficiency and effectiveness of the `MOIHT` and `SFSD` approaches presented in Sections 8.3.1 and 8.3.2 (Chapter 8) re-

[5]The implementation code of the `MOIHT` and `SFSD` methodologies can be found at `https://github.com/pierlumanzu/cc-moo` [78]

spectively.

In our numerical experience, we considered two classes of problems: cardinality-constrained quadratic problems and sparse logistic regression tasks.

The quadratic MOO problems, which often represent a useful test benchmark in optimization, have the form

$$\min_{x\in\mathbb{R}^n} \quad \tfrac{1}{2}\left(x^\top Q_1 x - c_1^\top x, x^\top Q_2 x - c_2^\top x\right)^\top \quad \text{s.t.} \quad \|x\|_0 \le s,$$

where $Q_1, Q_2 \in \mathbb{R}^{n\times n}$ are random positive semi-definite matrices and $c_1, c_2 \in \mathbb{R}^n$ are vectors whose values are randomly sampled in the range $[-1, 1)$. In the experiments, we varied the following problem parameters: the size $n \in \{10, 25, 50\}$; the condition number of the matrices $\kappa \in \{1, 10, 100\}$; the cardinality upper bound s. In particular, the latter was set in the following way: for $n = 10$, $s \in \{2, 5, 8\}$; for $n = 25$, $s \in \{5, 10, 20\}$; for $n = 50$, $s \in \{5, 15, 30\}$. Moreover, we used 3 different seeds for the pseudo-random number generator, thus leading to a total of 81 quadratic problems. For each instance, Q_1 and Q_2 are characterized by the same condition number, i.e., $L(f_1) = L(f_2) = \kappa$.

As for the sparse logistic regression problem [7, 20], it is a relevant task in machine and statistical learning. Given a dataset of N samples with n features $R = (r_1, \ldots, r_N)^\top \in \mathbb{R}^{N\times n}$ and N corresponding labels $\{t_1, \ldots, t_N\}$ belonging to $\{-1, 1\}$, the *regularized sparse logistic regression problem* is given by:

$$\min_{w\in\mathbb{R}^n} \ \tfrac{1}{N}\sum_{i=1}^{N} \log\left(1 + \exp\left(-t_i\left(w^\top r_i\right)\right)\right) + \tfrac{\lambda}{2}\|w\|^2 \quad \text{s.t.} \quad \|w\|_0 \le s,$$

where $\lambda \ge 0$. The logistic loss aims to fit the training data, while the regularization term helps to avoid overfitting. The two functions are clearly in contrast with each other. For our experiments, we employed the multi-objective reformulation considered in [57]:

$$\min_{w\in\mathbb{R}^n} \quad \left(\tfrac{1}{N}\sum_{i=1}^{N}\log(1+\exp(-t_i(w^\top r_i))),\ \tfrac{1}{2}\|w\|^2\right)^\top \quad \text{s.t.} \quad \|w\|_0 \le s.$$

For this problem, $L(F) = (\|R^\top R\| / N, 1)^\top$. The dataset suite we considered is composed of 7 binary classification datasets from the UCI Machine Learning Repository [31] (Table 9.15). We tested the algorithms on instances of the problem with $s \in \{2, 5, 8, 12, 20\}$. For each dataset, the samples with

missing values were removed. Moreover, the categorical variables were one-hot encoded, while the other ones were standardized to zero mean and unit standard deviation.

Dataset	N	n
Heart (Statlog)	270	25
Breast Cancer Wisconsin (Prognostic)	194	33
QSAR Biodegradation	1055	41
SPECTF heart	267	44
Spambase	4601	57
Optical recognition of handwritten digits	3823	62
Madelon	2000	500

Table 9.15: Datasets used for the experiments on sparse logistic regression. The number of features takes into account one-hot encoding of categorical ones.

Note that, for both classes of problems, in the following we will also consider solution approaches based on scalarization, i.e., tackling the problem $\min_{x\in\Omega} \; f_1(x) + \lambda f_2(x)$, where $\lambda \geq 0$. In the quadratic case, the problem can be solved by means of commercial solvers such as Gurobi, exploiting an `MIQP` reformulation. In the logistic regression case, we instead use the *Greedy Sparse-Simplex* (`GSS`) algorithm [3]. Note that, opposed to `MIQP` approach in quadratic problems, `GSS` is not guaranteed to produce a Pareto optimal solution.

As anticipated in Section 8.3.2.1, our `SFSD` methodology was tested taking as starting solutions the ones generated by the single-point methods mentioned above, i.e., `MOIHT`, `MOSPD`, `MOHyb`, `MIQP` and `GSS`.

Every execution had a time limit of 4 minutes. In particular, each single-point method was tested in a multi-start fashion: it had to process as many input points as possible within 2 minutes; in the remaining time, the `MOSD` procedure (Section 3.1.1) was employed as a refiner, starting at each returned point and keeping fixed its zero variables so that the cardinality constraint was kept valid. In `SFSD`, we set a time limit of 2 minutes for both phases of the algorithm.

For `MOIHT`, `MOSPD` and `MOHyb`, we considered $2n$ initial solutions randomly sampled from a box ($[-2, 2]^n$ for the quadratic problems; $[0, 1]^n$ for logistic regression). In order to be feasible, each initial point is first projected onto

Ω. These algorithms were executed 5 times with different seeds for the pseudo-random number generator to reduce the sensibility from the random initialization. The five generated fronts were then compared based on the *Purity* metric and only the best and worst ones were chosen for the comparisons. The scalarization-based approaches were run once considering $2n$ values for λ, i.e., $\lambda \in \{2^{i+\frac{1}{2}} \mid i \in \mathbb{Z},\ i \in [-n, n)\}$, and starting at the initial solution $\mathbf{0}_n \in \Omega$.

9.6.1 Quadratic Problems

In this section, we report the results on the cardinality-constrained quadratic problems. As for the algorithms parameters, based on some preliminary experiments not reported here for the sake of brevity, we set: $\varepsilon = 10^{-7}$, $L = 1.1\kappa$ for MOIHT; $\tau_{k+1} = 1.5\tau_k$, $\varepsilon_0 = 10^{-2}$, $\varepsilon_{k+1} = 0.9\varepsilon_k$ and $\|x_{k+1} - y_{k+1}\| \leq 10^{-3}$ as stopping condition for MOSPD; the Pareto stationarity approximation degree $\varepsilon = 10^{-7}$ for MOSD (see Definition 2.4 and Section 3.1). The parameters for the Armijo-type line searches are already listed in Section 9.1.2. Possible values for the MOSPD parameter τ_0 are discussed in the next section. The parameters choices for MOIHT and MOSPD were also used in MOHyb.

9.6.1.1 Preliminary Assessment of MOIHT, MOSPD and MOHyb

We start analyzing the effectiveness of MOIHT, MOSPD and MOHyb, comparing them in Figure 9.20 on a selection of quadratic problems. In order to show the differences among the algorithms as clearly as possible, only for this experiment, we considered a single run where the methods took as input the same 25 randomly extracted initial points. Moreover, we set no time limit, so that all the algorithms could process each initial solution until the respective stopping criteria were met.

The MOSPD and MOHyb performance was investigated for values for $\tau_0 \in \{1, 100\}$ (results for $\tau_0 = 100$ are shown in the left column of the figure, $\tau_0 = 1$ on the right). The black dots indicate the reference front: the latter is obtained combining the fronts retrieved by running SFSD with all the proposed initialization strategies and discarding the dominated solutions.

In well-conditioned problems ($\kappa = 1$), the MOIHT algorithm performs well, reaching solutions that belong to the reference front. As for MOSPD, the results with $\tau_0 = 100$ are worse, with MOSPD obtaining solutions far from the reference front. The situation is further stressed as the problem dimension n

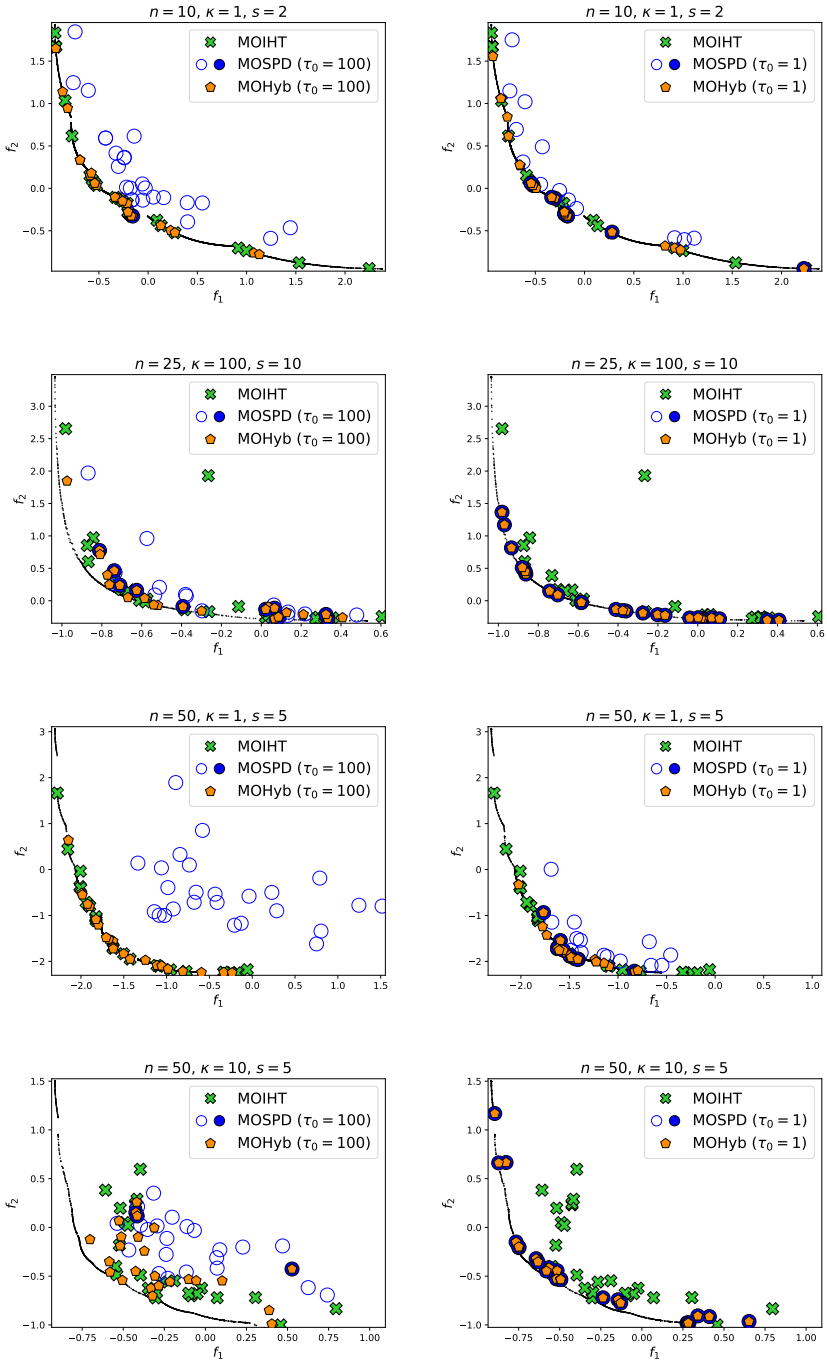


Figure 9.20: Results achieved by `MOIHT`, `MOSPD` and `MOHyb` with $\tau_0 \in \{1, 100\}$, starting at 25 random initial solutions, on a selection of quadratic problems. The filled markers denote L-stationary solutions ($L = 1.1\kappa$). The small black dots form the reference front.

grows. This sounds reasonable, since setting $\tau_0 = 100$ in Penalty Decomposition schemes binds the variables close to the initial feasible solution and, as a consequence, prevents from exploiting the exploration capabilities of MOSPD. A better choice for τ_0 ($\tau_0 = 1$) improves the performance of the algorithm, although MOIHT still performs better. This result is somewhat in line with the theory: MOIHT generates L-stationary points, whereas MOSPD converges to solutions only guaranteed to satisfy the (weaker) MOLZ conditions. In this scenario, MOHyb inherits the effectiveness of MOIHT: regardless the value for τ_0, it succeeded in getting solutions of the reference front.

In ill-conditioned problems ($\kappa > 1$), the MOIHT performance gets worse: the method struggled in reaching the reference front. This might be explained by the larger values of L that have to be used with these problems ($L = 1.1\kappa$). As the value of L grows, the L-stationarity condition does not provide enough information on the quality of the solution support set. As a consequence, MOIHT can end up in many L-stationary points with "bad" support, i.e., far from the actual Pareto front of the problem. MOSPD with $\tau_0 = 1$ obtained better solutions in these cases. Employing lower values for τ_0, the approach is initially allowed to search for a good point minimizing $F(\cdot)$: this feature can be crucial to avoid a large portion of "bad" L-stationary points and to reach solutions in the reference front. MOHyb ($\tau_0 = 1$) proved to be effective in these scenarios too. The hybrid approach, in these ill-conditioned cases, profited from the exploration capabilities of MOSPD, reaching the same solutions. However, like in the well-conditioned case, MOHyb also proved to be less sensitive than MOSPD w.r.t. the value of τ_0, taking advantage of the MOIHT mechanisms to reach at least L-stationary solutions when $\tau_0 = 100$.

9.6.1.2 Evaluation of the SFSD Methodology

We start the analysis of the SFSD algorithm performance on the quadratic problems through Figure 9.21, where we show how the front descent phase allows to improve the results of basic multi-start approaches corresponding to phase one, i.e., MOIHT, MOSPD, MOHyb and MIQP. According to the results of Section 9.6.1.1, for MOSPD and MOHyb, we set $\tau_0 = 1$.

As anticipated in Section 8.3.2, in cardinality-constrained MOO the Pareto fronts are typically irregular and made up of several smooth parts. The plots in the figure perfectly reflect these characteristics: each front portion can indeed be associated with a specific support set. Starting from the solutions generated by the single-point methods, the SFSD methodology proves to be

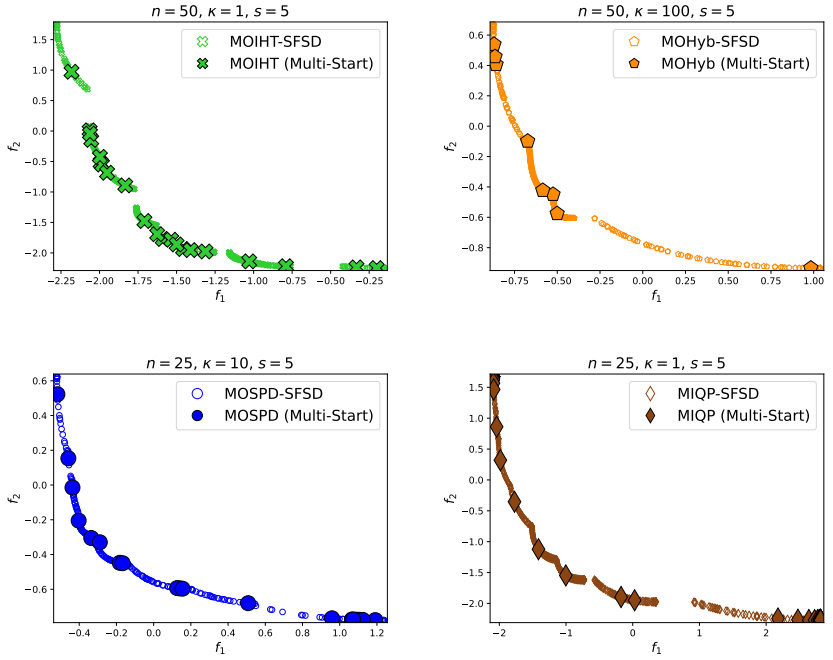


Figure 9.21: Results of `SFSD` phase two compared to simple `MOSD` refinement of solutions retrieved in phase one. We show one example instance for each considered multi-start/phase one strategy.

effective in exhaustively spanning each portion associated with a support set. This feature allowed our novel front-oriented approach to identify regions of the Pareto front that would have otherwise been hardly covered with the multi-start strategy. As mentioned in Section 8.3.2, to the best of our knowledge, `SFSD` is the first front-oriented approach for cardinality-constrained MOO. In the absence of other specialized algorithms, it is difficult to quantitatively assess the potential of our algorithm and we need to resort to the visual inspection of the solutions. In the rest of the section, we then focus our attention on the different options we outlined for the phase one of the `SFSD` algorithm, comparing its performance as the initialization strategy varies. The comparisons were made by means of the performance profiles (Figure 9.22) on the entire benchmark of quadratic problems.

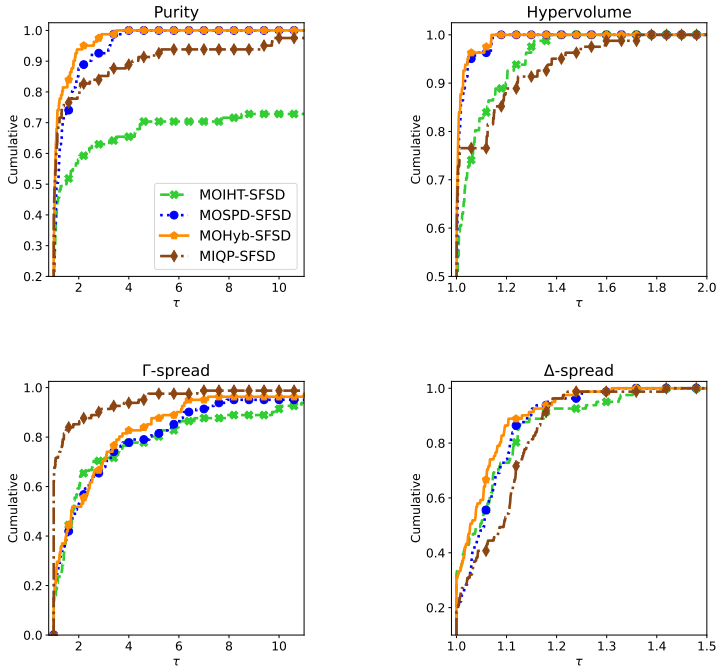


Figure 9.22: Performance profiles for SFSD with different initialization strategies, i.e., MOIHT, MOSPD, MOHyb (best executions w.r.t. *Purity*) and MIQP on the quadratic problems.

Looking at the *Purity* and the *Hypervolume* metrics, SFSD resulted to be more robust with MOHyb as initialization strategy instead of MOIHT and MOSPD. These results reflect the behavior of the three single-point algorithms already shown in Figure 9.20: while MOIHT resulted to be more effective on the well-conditioned problems, MOSPD, with a right choice for the value for τ_0, performed better on the (larger) set of ill-conditioned problems; MOHyb, inheriting the mechanisms of both, managed to obtain good results on problems of both types. As for Γ–*spread*, MIQP proves to be more capable than the other single-point methods in generating solutions in the extreme regions of the objectives space, and this fact allowed SFSD to get wider front reconstructions. The performance of our front-oriented approach with MOHyb, MOIHT and MOSPD employed in the phase one was quite similar in this sce-

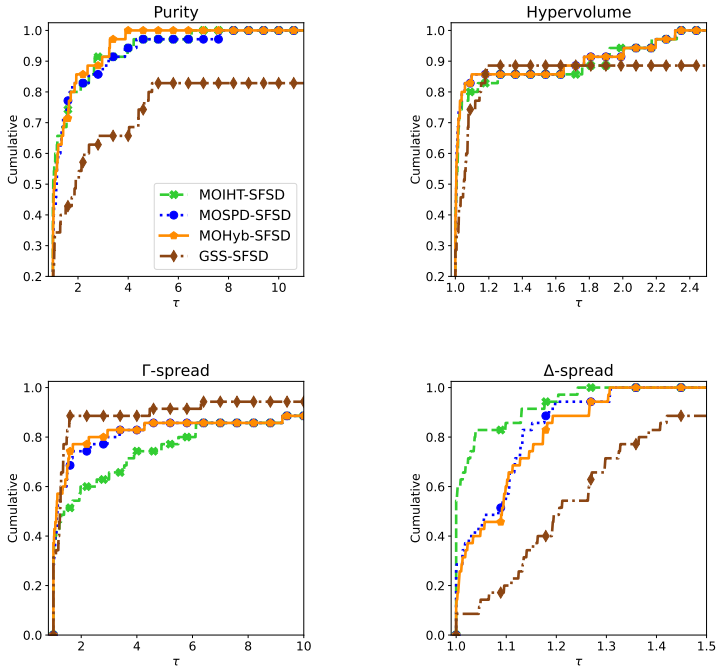


Figure 9.23: Performance profiles for `SFSD` with different initialization strategies, i.e., `MOIHT`, `MOSPD`, `MOHyb` (best executions w.r.t. *Purity*) and `GSS` on 35 logistic regression problems.

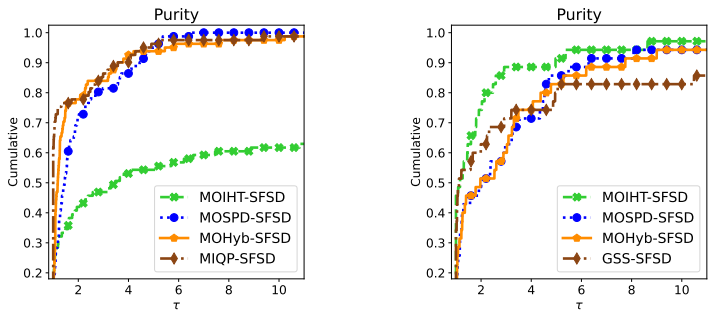
nario. Regarding the Δ–*spread* metric, i.e., uniformity of the Pareto front approximation, all the initialization strategies led to comparable results.

9.6.2 Logistic Regression

In this last section, we analyze the performance profiles (Figure 9.23) on the logistic regression problems for `SFSD` with the different possible choices for the first phase of the algorithm. The values for the parameters of the algorithms were again chosen based on preliminary experiments which are not reported for the sake of brevity. In particular, we set: $L = 1.1 \max\{L(f_1), L(f_2)\}$ for `MOIHT`; $\varepsilon = 10^{-7}$ for both `MOIHT` and `MOSD`; $\tau_0 = 1$, $\tau_{k+1} = 1.3\tau_k$, $\varepsilon_0 = 10^{-5}$, $\varepsilon_{k+1} = 0.9\varepsilon_k$ and $\|x_{k+1} - y_{k+1}\| \leq 10^{-3}$ as stopping condition for `MOSPD`.

Again, the parameters for MOIHT and MOSPD were also employed in MOHyb. Finally, since the objective functions have different scales, similarly to what is done in [57], when computing the *spread* metrics we considered the logarithm (base 10) of the $f_2(\cdot)$ values and, then, re-scaled both $f_1(\cdot)$ and $f_2(\cdot)$ to have values in $[0, 1]$.

With respect to the *Purity* and the *Hypervolume* metrics, SFSD resulted to be more robust with MOIHT, MOSPD and MOHyb as initialization strategies, with MOHyb appearing to be slightly superior. As for the Γ*–spread* metric, GSS was the best algorithm for the SFSD phase one. However, although SFSD, equipped with this setting, was effective in reaching remote regions of the objectives space, it struggled to obtain uniform Pareto front approximations and, thus, to obtain good Δ*–spread* values. As for this last metric, using as initialization strategy MOIHT proved to be a better choice.



(a) Quadratic problems (b) Logistic regression problems

Figure 9.24: Performance profiles for SFSD with different initialization strategies, i.e., MOIHT, MOSPD, MOHyb (worst executions w.r.t. *Purity*) and MIQP/GSS.

Remark 9.1. In the previous sections, we considered the best executions of SFSD equipped with MOIHT, MOSPD and MOHyb, and we compared them with the deterministic outputs of our front-oriented algorithm when MIQP/GSS was employed in the phase one. Thus, for the sake of completeness, in Figure 9.24 we report the performance profiles w.r.t. the *Purity* metric obtained considering the worst runs. Comparing these performance profiles with the ones in Figures 9.22-9.23, we observe only slight decreases in the performance of the non-deterministic strategies.

Chapter 10

Conclusions

In this thesis, we considered a various set of multi-objective optimization (MOO) problems. In particular, the set includes unconstrained, box constrained, convex constrained and cardinality-constrained instances.

As first contribution of the dissertation, we presented an overview, as general as possible, of the main theoretical MOO concepts, with a focus on the ones concerning well-known gradient-based methodologies from the MOO literature. In particular, we proposed: a general formulation for the search (descent) direction problem; a framework for the gradient-based approaches. For both of them, we reported the main properties; moreover, we showed how they can be reduced to well-known schemes.

In the MOO context, reconstructing the Pareto front of the problem could be much more helpful than returning a single solution: in this way, having to choose from multiple solutions representing different tradeoffs of the objective functions, the final user would be free to choose a posteriori the best one for their scope. In the MOO literature, Pareto front reconstruction has been mainly addressed through evolutionary/derivative-free algorithms, while, to date, few first-order approaches have been introduced and analyzed. Most of the proposed gradient-based methodologies are indeed "single-point", i.e., designed to return a single solution. However, running such algorithms in a multi-start fashion may be ineffective to generate accurate and uniform Pareto front approximations, both from a conceptual and practical point of view.

In this thesis, we then focused on proposing innovative gradient-based methodologies designed to approximate, as accurately as possible, the Pareto

front of the considered problems. Among the employed tools, the common and partial descent directions certainly stand out, being potentially crucial to obtain convergence and exploration of the objectives space at the same time. For each approach, we reported and described the algorithmic scheme. In addition, each one was theoretically analyzed in order to state its characteristic properties, including the convergence one(s) to feasible points satisfying necessary optimality conditions.

From an experimental point of view, our methods were tested by means of thorough computational experiments, where they were compared with state-of-the-art approaches from the MOO literature. The results let us to conclude that our algorithms are effective and efficient w.r.t. the competitors in retrieving accurate, wide and uniform Pareto front approximations.

Finally, during the development of the presented works, some possible future research directions arose. We conclude the thesis reporting some of the most prominent ones.

- An extension of our memetic approach `NSMA` (Chapter 4) to be used in more general settings, such as multi-objective optimization problems with general convex/non-convex constraints.
- The development of an extension of our limited memory Quasi-Newton approach (Chapter 5) for box-constrained MOO problems, taking inspiration by `L-BFGS-B` [108], the well-known `L-BFGS` variant for single-objective optimization problems with bound constraints.
- The employment of our *Improved Front Steepest Descent* approach (Chapter 6) within memetic procedures, such as `NSMA`, for global multi-objective optimization.
- A possible variant of `FRONT-ALAMO` (Chapter 7) to deal with MOO problems with general, possibly non-convex, constraints.
- The extension of the theoretical results and the algorithms presented in Chapter 8 to handle additional constraints other than the cardinality one.

Appendix A

The Front Projected Gradient Algorithm

In this appendix, we describe an adaptation of the FSD method [23] (Algorithm 3.4) for box-constrained MOO problems, which we call *Front Projected Gradient* (FPG). We initially report the scheme of the new adaptation. Then, in the remainder of the appendix, we provide a rigorous theoretical analysis.

A.1 Algorithmic Scheme

We report the scheme of FPG in Algorithm A.1. The noteworthy differences w.r.t. the original approach are the following:

- the initial set X^0 is composed by *feasible* non-dominated points, i.e., $X^0 \subset \Omega$, with $\Omega = \{x \in \mathbb{R}^n \mid x \in [l, u], l, u \in \mathbb{R}^n \text{ s.t. } l \leq u\}$;
- the direction at the solution x_c is found solving an instance of Problem (A.1); by Problem (2.3) and Table 2.1, it is trivial to see that if $\theta_{\mathcal{I}}^{CS}(x_c) < 0$ thus $\theta_c^{\mathcal{I}} < 0$, i.e., $d_c^{\mathcal{I}}$ is a feasible and descent direction at x_c;
- FPG employs the *Bound-constrained Front Armijo-Type Line Search* (B-FALS), which we report in Algorithm A.2; the only added requirement w.r.t. FALS (Algorithm 3.3) is that the step size must lead to a point that is also feasible.

Algorithm A.1: Front Projected Gradient (`FPG`)

Input: $F : \mathbb{R}^n \to \mathbb{R}^m$, Ω feasible closed and convex set, $X^0 \subset \Omega$ set of mutually non-dominated points w.r.t. $F(\cdot)$.
$k = 0$
while *a stopping criterion is not satisfied* **do**
  $\hat{X}^k = X^k$
  forall $x_c \in X^k$ **do**
    forall $\mathcal{I} \subseteq \{1, \ldots, m\}$ *such that*
    - $x_c \in \hat{X}^k_{\mathcal{I}}$ *and*
    - $\theta^{CS}_{\mathcal{I}}(x_c) < 0$

    do
      Compute

$$d^{\mathcal{I}}_c = \arg\min_{d \in \mathbb{R}^n} \max_{j \in \mathcal{I}} \nabla f_j(x_c)^\top d + \frac{1}{2}\|d\|^2 \quad \text{s.t. } x_c + d \in \Omega \tag{A.1}$$

      Let $\theta^{\mathcal{I}}_c$ the optimal value of Problem (A.1) at x_c
      $\alpha = \texttt{B-FALS}(F(\cdot), \Omega, \mathcal{I}, \hat{X}^k_{\mathcal{I}}, x_c, d^{\mathcal{I}}_c, \theta^{\mathcal{I}}_c)$
      $\hat{X}^k = \left(\hat{X}^k \cup \left\{x_c + \alpha d^{\mathcal{I}}_c\right\}\right) \setminus \left\{y \in \hat{X}^k \mid F\left(x_c + \alpha d^{\mathcal{I}}_c\right) \lneqq F(y)\right\}$
  $X^{k+1} = \hat{X}^k$
  $k = k + 1$
return X^k

In the remainder of the appendix, we state some properties of both the methodologies.

A.2 Algorithm Analysis

We begin the analysis showing that `B-FALS` terminates in a finite number of iterations.

Proposition A.1. *Let $\mathcal{I} \subseteq \{1, \ldots, m\}$, $x_c \in X^k_{\mathcal{I}}$ be such that $\theta^{\mathcal{I}}_c < 0$, i.e.,*

Algorithm A.2: Bound-constrained Front Armijo-type Line Search (B-FALS)

Input: $F : \mathbb{R}^n \to \mathbb{R}^m$, Ω feasible closed and convex set, $\mathcal{I} \subseteq \{1, \ldots, m\}$, $X_{\mathcal{I}}^k \subset \Omega$ set of mutually non-dominated points w.r.t. $F_{\mathcal{I}}(\cdot)$, $x_c \in X_{\mathcal{I}}^k$, $d_c^{\mathcal{I}}$ feasible descent direction, $\theta_c^{\mathcal{I}} \in \mathbb{R}$, $\alpha_0 > 0$, $\delta \in (0,1)$, $\gamma \in (0,1)$.
$\alpha = \alpha_0$
while $x_c + \alpha d_c^{\mathcal{I}} \notin \Omega \vee \exists y \in X_{\mathcal{I}}^k$ *s.t.* $F_{\mathcal{I}}(y) + \mathbf{1}_{|\mathcal{I}|}\gamma\alpha\theta_c^{\mathcal{I}} < F_{\mathcal{I}}(x_c + \alpha d_c^{\mathcal{I}})$ **do**
  $\alpha = \delta\alpha$
return α

$d_c^{\mathcal{I}}$ defined as in (A.1) *exists such that*

$$\nabla f_j(x_c)^\top d_c^{\mathcal{I}} + \frac{1}{2}\|d_c^{\mathcal{I}}\|^2 < 0, \qquad \forall j \in \mathcal{I}.$$

Then $\exists \bar{\alpha} > 0$ *such that*

$$x_c + \bar{\alpha} d_c^{\mathcal{I}} \in \Omega$$

and

$$F_{\mathcal{I}}(y) + \mathbf{1}_{|\mathcal{I}|}\gamma\bar{\alpha}\theta_c^{\mathcal{I}} \not< F_{\mathcal{I}}(x_c + \bar{\alpha} d_c^{\mathcal{I}}), \qquad \forall y \in X_{\mathcal{I}}^k,$$

i.e., the while loop of B-FALS *terminates in a finite number* $\bar{h}$ *of iterations, returning a value* $\bar{\alpha} = \delta^{\bar{h}}\alpha_0$. *Furthermore, the produced point* $x_c + \bar{\alpha} d_c^{\mathcal{I}}$ *is not dominated with respect to the set* X^k.

Proof. Assume by contradiction that the thesis is false. Then the algorithm produces an infinite sequence $\{\delta^h \alpha_0\}$ such that, for all h, either

$$x_c + \delta^h \alpha_0 d_c^{\mathcal{I}} \notin \Omega$$

or a point $y_h \in X_{\mathcal{I}}^k$ exists such that

$$F_{\mathcal{I}}(y_h) + \mathbf{1}_{|\mathcal{I}|}\gamma\delta^h\alpha_0\theta_c^{\mathcal{I}} < F_{\mathcal{I}}(x_c + \delta^h\alpha_0 d_c^{\mathcal{I}}). \tag{A.2}$$

By the convexity of Ω and the definition of the direction $d_c^{\mathcal{I}}$ (A.1), since $\delta^h\alpha_0 \to 0$ as $h \to \infty$, for h sufficiently large the point $x_c + \delta^h\alpha_0 d_c^{\mathcal{I}}$ is feasible and thus condition (A.2) holds. Then, following the proof of Proposition 4 in [23], we can prove the thesis. □

Regarding FPG, the first property we prove is about the feasibility of the points produced by the algorithm.

Proposition A.2. *Let $\{X^k\}$ be the sequence of sets of points generated by FPG. Then, for all k, every point x_c in the set X^k is feasible for Problem* (2.1).

Proof. The proof is straightforward. First of all, the initial set X^0 is composed by feasible points. New solutions are only added through Line 11. Considering the convexity of Ω and the definition of the direction $d_c^{\mathcal{I}}$ (A.1) for any $\mathcal{I} \subseteq \{1, \ldots, m\}$, and reminding the stopping criteria of B-FALS, these new points are contained in Ω and, therefore, they are feasible for Problem (2.1). □

Finally, we are ready to state the convergence property of FPG. In order to prove it, we make use of Assumption 3.2 and of the concept of linked sequence (Definition 3.1).

Proposition A.3. *Let us assume that Assumption 3.2 holds. Let $\{X^k\}$ be the sequence of sets of non-dominated points w.r.t. $F(\cdot)$ produced by FPG. Let $\{x_{j_k}\}$ be a linked sequence, then it admits limit points and every limit point is Pareto-stationary for Problem* (2.1).

Proof. The proof is almost identical to the one of Proposition 5 in [23]. There is mainly one difference. After proving that

$$\lim_{\substack{k \to \infty \\ k \in K}} \alpha_{j_{k+1}} = 0 \tag{A.3}$$

[23, Equation 22], where K indicates a subsequence, the FSD authors consider sufficiently large values of k such that $\alpha_{j_{k+1}} < \alpha_0$. In this case, the steps of FALS (Algorithm 3.3) and the definition of X^k, $k \in K$, imply that there exists $y_k \in X^k$ such that

$$F_{\mathcal{I}_k}(y_k) + \mathbf{1}_{|\mathcal{I}_k|}\gamma\frac{\alpha_{j_{k+1}}}{\delta}\theta^{SD}_{\mathcal{I}_k}(x_{j_k}) < F_{\mathcal{I}_k}\left(x_{j_k} + \frac{\alpha_{j_{k+1}}}{\delta}v^{SD}_{\mathcal{I}_k}(x_{j_k})\right).$$

With respect to FALS, B-FALS has an additional stopping criterion: the step size must lead to a point that is feasible for Problem (2.1). In this context, $\alpha_{j_{k+1}}/\delta \leq \alpha_0$ might not have been selected because the point $x_{j_k} + (\alpha_{j_{k+1}}/\delta)d_{j_k}^{\mathcal{I}_k} \notin \Omega$. However, through a little modification, we can handle this additional stopping criterion.

First of all, Equation (A.3) still holds: the proof of this statement is the same provided in [23]. Then, we can consider sufficiently large values of k such that

$$\alpha_{j_{k+1}} < \frac{\alpha_{j_{k+1}}}{\delta} \leq 1.$$

In this way, by the convexity of Ω and the definition of the direction $d_{j_k}^{\mathcal{I}_k}$ (A.1), the points produced by the two step sizes are feasible, i.e., the `B-FALS` feasibility stopping criterion is satisfied. Then, the steps of `B-FALS` and the definition of X^k, $k \in K$, imply that there exists $y_k \in X^k$ such that

$$F_{\mathcal{I}_k}(y_k) + \mathbf{1}_{|\mathcal{I}_k|}\gamma\frac{\alpha_{j_{k+1}}}{\delta}\theta_{j_k}^{\mathcal{I}_k} < F_{\mathcal{I}_k}\left(x_{j_k} + \frac{\alpha_{j_{k+1}}}{\delta}d_{j_k}^{\mathcal{I}_k}\right).$$

From this point forward, we can follow the remainder of the proof of Proposition 5 in [23] in order to prove the thesis. □

Appendix B

Analysis of the Search Direction Problem

In this appendix, we propose a further analysis on Problem (2.3) with $\Omega = \mathbb{R}^n$. In particular, we show the steps to get the dual form of the problem, along with characteristics typical of its optimal solutions and Lagrange multipliers.

As shown in Section 2.2, given $\bar{x} \in \Omega$, Problem (2.3) can be re-written as

$$\min_{\substack{\beta \in \mathbb{R} \\ d \in \mathcal{D}(\bar{x})}} \quad \beta \quad \text{s.t. } \nabla f_j(\bar{x})^\top d + \frac{1}{2} d^\top M_j(\bar{x}) d - \beta \leq 0, \quad \forall j \in \{1, \dots, m\}.$$

In this scenario, the Lagrangian function is of the following form:

$$L(\beta, d, \lambda) = \beta + \sum_{j=1}^{m} \lambda_j \left[\nabla f_j(\bar{x})^\top d + \frac{1}{2} d^\top M_j(\bar{x}) d - \beta \right],$$

where $\lambda_1, \dots, \lambda_m$ are the Lagrange multipliers. Denoting by $\lambda \in \mathbb{R}^m$ the vector of all the Lagrange multipliers, we also introduce the dual problem:

$$\max_{\lambda \in \mathbb{R}^m} \inf_{\substack{\beta \in \mathbb{R} \\ d \in \mathcal{D}(\bar{x})}} L(\beta, d, \lambda) \quad \text{s.t. } \lambda \geq \mathbf{0}_m.$$

Now, let us consider $\Omega = \mathbb{R}^n$, along with the positive definiteness of the matrices $M_1(\cdot), \dots, M_m(\cdot)$, as, in this special case, we can address some features of the dual form and the solutions more explicitly. In Section 2.2, we indeed state that, if $M_j(\cdot) \succ 0 \ \forall j \in \{1, \dots, m\}$, then Problem (2.3) has

a unique solution. Moreover, in such scenario, we have that the problem is convex and has a Slater point, i.e., $(\beta, d) = (1, \mathbf{0}_n)$. Then, strong duality holds and the Karush-Kuhn-Tucker (KKT) conditions are sufficient and necessary for optimality:

$$\frac{\partial L(\beta, d, \lambda)}{\partial \beta} = 1 - \sum_{j=1}^{m} \lambda_j = 0, \qquad \frac{\partial L(\beta, d, \lambda)}{\partial d} = \sum_{j=1}^{m} \lambda_j \left[\nabla f_j(\bar{x}) + M_j(\bar{x}) d\right] = \mathbf{0}_n,$$

which leads to

$$\sum_{j=1}^{m} \lambda_j = 1, \qquad d = -\left[\sum_{j=1}^{m} \lambda_j M_j(\bar{x})\right]^{-1} J_F(\bar{x})^{\top} \lambda. \tag{B.1}$$

Considering Equation (B.1), given $\lambda \geq \mathbf{0}_m$, we can state that in MOO the search direction depends on convex combinations of both the matrices $M_1(\cdot), \ldots, M_m(\cdot)$ and the gradients.

In scalar optimization ($m = 1$, $f(\cdot)$ indicates the objective function), depending on the form of the matrix $M_1(\cdot)$, the second formula of (B.1) reduces to standard search directions:

- if $M_1(\cdot) = I_n$, the formula reduces to the steepest descent direction $d = -\nabla f(\bar{x})$;
- if $M_1(\cdot) = \nabla^2 f(\cdot)$, the formula reduces to the Newton direction $d = -[\nabla^2 f(\bar{x})]^{-1} \nabla f(\bar{x})$;
- if $M_1(\cdot) = B$, with B being the Quasi-Newton approximation matrix, the formula reduces to the Quasi-Newton direction $d = -B^{-1} \nabla f(\bar{x})$;

Equation (B.1) also leads to re-write the dual function; in particular, β disappears since $\sum_{j=1}^{m} \lambda_j = 1$, while d is substituted:

$$\inf_{\substack{\beta \in \mathbb{R} \\ d \in \mathbb{R}^n}} L(\beta, d, \lambda) = -\frac{1}{2} \lambda^{\top} J_F(\bar{x}) \left[\sum_{j=1}^{m} \lambda_j M_j(\bar{x})\right]^{-1} J_F(\bar{x})^{\top} \lambda$$

Using this last equation, we can finally retrieve the final form of the dual problem:

$$\begin{aligned} \max_{\lambda \in \mathbb{R}^m} \; & -\frac{1}{2} \lambda^{\top} J_F(\bar{x}) \left[\sum_{j=1}^{m} \lambda_j M_j(\bar{x})\right]^{-1} J_F(\bar{x})^{\top} \lambda \\ \text{s.t.} \; & \sum_{j=1}^{m} \lambda_j = 1, \qquad \lambda \geq \mathbf{0}_m. \end{aligned}$$

Appendix C

Supplementary Mathematical Proofs

In this appendix, we provide the proofs of propositions and lemmas that did not find space in the main body of the thesis.

Proposition 5.1. *Considering a generic iteration k of Algorithm 5.1, let $x_k \in \mathbb{R}^n$, $v^{LM}(x_k) \in \mathbb{R}^n$ be a direction such that $D\left(x_k, v^{LM}(x_k)\right) < 0$, $\alpha_k > 0$ be a step size along $v^{LM}(x_k)$ and $\lambda^{LM}(x_k)$ be the Lagrange multipliers vector obtained solving Problem (5.2). If ρ^k is updated by (5.7), then ρ^k is positive.*

Proof. First, the case $s_k^\top u_k > 0$ is trivial.

Then, let $s_k^\top u_k \leq 0$ and let us consider $D\left(x_{k+1}, s_k\right) - \nabla f_j\left(x_k\right)^\top s_k$, with $j \in \{1, \ldots, m\}$. Considering that $D(x_k, s_k) \geq \nabla f_j(x_k)^\top s_k$ by definition of $D(\cdot, \cdot)$ and Equation (3.1), we have that

$$\begin{aligned} & D\left(x_{k+1}, s_k\right) - \nabla f_j\left(x_k\right)^\top s_k \\ & \qquad \geq D\left(x_{k+1}, s_k\right) - D\left(x_k, s_k\right) \\ & \qquad = D\left(x_{k+1}, \alpha_k v^{LM}(x_k)\right) - D\left(x_k, \alpha_k v^{LM}(x_k)\right). \end{aligned} \tag{C.1}$$

In Algorithm 5.1, the Wolfe conditions are imposed at each iteration k (Line 7): in particular, $D\left(x_{k+1}, v^{LM}(x_k)\right) \geq \sigma D\left(x_k, v^{LM}(x_k)\right)$. Then, also con-

sidering Lemma 2.3, we obtain that

$$\begin{aligned} D\left(x_{k+1}, \alpha_k v^{LM}(x_k)\right) - D\left(x_k, \alpha_k v^{LM}(x_k)\right) & \\ = \alpha_k \left[D\left(x_{k+1}, v^{LM}(x_k)\right) - D\left(x_k, v^{LM}(x_k)\right)\right] & \quad \text{(C.2)} \\ \geq \alpha_k \left(\sigma - 1\right) D\left(x_k, v^{LM}(x_k)\right) > 0, & \end{aligned}$$

where the last inequality comes from the fact that $\alpha_k > 0$, $\sigma - 1 < 0$ and $D\left(x_k, v^{LM}(x_k)\right) < 0$. Using (C.1) and (C.2), we conclude that

$$D\left(x_{k+1}, s_k\right) - \nabla f_j\left(x_k\right)^\top s_k > 0. \tag{C.3}$$

Since we have considered a generic $j \in \{1, \ldots, m\}$, it trivially follows that this last equation is verified for all j.

Now, given (2.6), which is valid for the Lagrange multipliers vector $\lambda^{LM}\left(x_k\right)$, and (C.3), we can state that

$$\sum_{j=1}^{m} \lambda_j^{LM}(x_k) \left[D\left(x_{k+1}, s_k\right) - \nabla f_j\left(x_k\right)^\top s_k\right] > 0.$$

Then, if the second formula in (5.7) is used, $\rho^k > 0$. □

Proposition 5.2. *Let Assumption 3.1 hold with $\Omega = \mathbb{R}^n$. Let $x_k \in \mathbb{R}^n$ and assume that $d_k \in \mathbb{R}^n$ is a direction such that $D(x_k, d_k) < 0$, $\gamma \in (0, 1/2)$ and $\sigma \in (\gamma, 1)$. Then, there exists an interval of values $[\alpha_l, \alpha_u]$, with $0 < \alpha_l < \alpha_u$, such that for all $\alpha \in [\alpha_l, \alpha_u]$ Equations* (5.8) *and* (5.9) *hold.*

Proof. Given $F(\cdot)$ continuously differentiable and d_k descent direction for $F(\cdot)$ at x_k and recalling Proposition 3.2 with $\Omega = \mathbb{R}^n$, we can state that there exists $\varepsilon > 0$ such that, for all $\alpha \in (0, \varepsilon]$, we have

$$F\left(x_k + \alpha d_k\right) < F\left(x_k\right) + \gamma \alpha J_F(x_k) d_k \leq F\left(x_k\right) + \mathbf{1}_m \gamma \alpha D\left(x_k, d_k\right), \tag{C.4}$$

where the last inequality follows from Remark 3.1. We now assume, by contradiction, that for all $\alpha > 0$ we have

$$F\left(x_k + \alpha d_k\right) < F\left(x_k\right) + \mathbf{1}_m \gamma \alpha D\left(x_k, d_k\right).$$

This last result indicates that we have $\{x_k + \alpha d_k \mid \alpha > 0\} \subseteq \mathcal{L}_F\left(F(x_k)\right)$, which is absurd since Assumption 3.1 holds. Therefore, there exists $\hat{\alpha} > \varepsilon$ and $\hat{j} \in \{1, \ldots, m\}$ such that

$$f_{\hat{j}}\left(x_k + \hat{\alpha} d_k\right) = f_{\hat{j}}\left(x_k\right) + \gamma \hat{\alpha} D\left(x_k, d_k\right) \tag{C.5}$$

and, by the continuity of $F(\cdot)$, equation (C.4) holds for all $\alpha < \hat{\alpha}$.

Using the Mean-value Theorem, we have that

$$f_{\hat{j}}\left(x_k + \hat{\alpha} d_k\right) = f_{\hat{j}}\left(x_k\right) + \hat{\alpha} \nabla f_{\hat{j}}\left(x_k + t\hat{\alpha} d_k\right)^{\top} d_k, \tag{C.6}$$

with $t \in (0,1)$. Combining (C.5) and (C.6) we get

$$\nabla f_{\hat{j}}\left(x_k + t\hat{\alpha} d_k\right)^{\top} d_k = \gamma D\left(x_k, d_k\right) > \sigma D\left(x_k, d_k\right), \tag{C.7}$$

where the last inequality follows from the fact that $\sigma > \gamma$ and $D\left(x_k, d_k\right) < 0$.

By definition of $D(\cdot,\cdot)$, we have the following condition: $D\left(x_k + t\hat{\alpha} d_k, d_k\right) \geq \nabla f_{\hat{j}}\left(x_k + t\hat{\alpha} d_k\right)^{\top} d_k$. Using the latter and (C.7), we obtain that

$$D\left(x_k + t\hat{\alpha} d_k, d_k\right) > \sigma D\left(x_k, d_k\right).$$

Then, by the continuity of $D(\cdot,\cdot)$, there exists an interval $[\alpha_l, \alpha_u] \subseteq (0, \hat{\alpha})$ such that, for all $\alpha \in [\alpha_l, \alpha_u]$, we have that $D\left(x_k + \alpha d_k, d_k\right) > \sigma D\left(x_k, d_k\right)$. Moreover, since for all $\alpha \in [0, \hat{\alpha}]$ Equation (5.8) holds and $[\alpha_l, \alpha_u] \subseteq (0, \hat{\alpha})$, the proof is complete. □

Lemma 5.1. *Consider a generic iteration t of Algorithm 5.3. Let $x_k \in \mathbb{R}^n$ and d_k be a direction such that $D(x_k, d_k) < 0$. Then, we have the following properties:*

1. *if $\alpha_u^t < \infty$, then*

$$\exists\, j\left(\alpha_u^t\right) \ \text{ s.t. } \ f_{j(\alpha_u^t)}(x_k + \alpha_u^t d_k) > f_{j(\alpha_u^t)}(x_k) + \gamma \alpha_u^t D(x_k, d_k); \tag{C.8}$$

2. *α_l^t is such that*

$$F(x_k + \alpha_l^t d_k) \leq F(x_k) + \mathbf{1}_m \gamma \alpha_l^t D(x_k, d_k), \tag{C.9}$$

$$D(x_k + \alpha_l^t d_k, d_k) < \sigma D(x_k, d_k). \tag{C.10}$$

Proof. 1. Since $\alpha_u^0 = \infty$ and $\alpha_u^t < \infty$, the interval upper bound has been updated at least once though Line 5. Let $\bar{t}$ be the iteration in which this update takes place: it follows that $0 \leq \bar{t} < t$. In this case, we have that

$$\exists\, j\left(\alpha^{\bar{t}}\right) \in \{1, \ldots, m\} \ \text{ s.t. } \ f_{j\left(\alpha^{\bar{t}}\right)}(x_k + \alpha^{\bar{t}} d_k) > f_{j\left(\alpha^{\bar{t}}\right)}(x_k) + \gamma \alpha^{\bar{t}} D(x_k, d_k)$$

and $\alpha_u^{\bar{t}+1} = \alpha^{\bar{t}}$. Then, it trivially follows that Equation (C.8) holds for $\alpha_u^{\bar{t}+1}$.

Now, suppose that $\bar{t}+1 < t$ and consider the iteration $\bar{t}+1$. By the instructions of the algorithm, $\alpha_u^{\bar{t}+2}$ is updated either by $\alpha_u^{\bar{t}+2} = \alpha^{\bar{t}+1}$ (Line 5) or $\alpha_u^{\bar{t}+2} = \alpha_u^{\bar{t}+1}$ (Line 8). The first case is identical to the one of iteration $\bar{t}$. In the second case, since (C.8) is satisfied for $\alpha_u^{\bar{t}+1}$, it is also for $\alpha_u^{\bar{t}+2}$. For $t > \bar{t}+2$, the property follows by induction.

2. Given $\sigma \in (0,1)$ and $D(x_k, d_k) < 0$, it is easy to prove that (C.9) and (C.10) hold for $\alpha_l^0 = 0$.

   Then, consider the first iteration of Algorithm 5.3. Similarly to the upper bound, α_l^1 is set equal to either α_l^0 (Line 6) or α^0 (Line 10). The latter case occurs if $F(x_k + \alpha^0 d_k) \leq F(x_k) + \mathbf{1}_m \gamma \alpha^0 D(x_k, d_k)$ and $D(x_k + \alpha^0 d_k, d_k) < \sigma D(x_k, d_k)$. Thus, it simply follows that conditions (C.9) and (C.10) are satisfied in both cases.

   For $t > 1$, we get the thesis by induction.

□

Proposition 5.3. *Let Assumption 3.1 hold with $\Omega = \mathbb{R}^n$, $\delta \in [1/2, 1)$, $\eta > 1$ and let $\{\alpha_l^t, \alpha_u^t, \alpha^t\}$ be the sequence generated by Algorithm 5.3. Assume that:*

1. *$d_k \in \mathbb{R}^n$ is a descent direction for $F(\cdot)$ at $x_k \in \mathbb{R}^n$;*
2. *for all $t > 0$, the step size α^t is chosen so that*
   a. *if $\alpha_u^t = \infty$,*
   $$\alpha^t \geq \eta \max\left\{\alpha_l^t, \alpha^0\right\}, \tag{C.11}$$
   b. *if $\alpha_u^t < \infty$,*
   $$\max\left\{\left(\alpha^t - \alpha_l^t\right), \left(\alpha_u^t - \alpha^t\right)\right\} \leq \delta\left(\alpha_u^t - \alpha_l^t\right).$$

Then Algorithm 5.3 is well defined, i.e., it stops after a finite number of iterations returning a step size $\hat{\alpha}$ satisfying the Wolfe conditions for MOO.

Proof. By contradiction, we assume that the thesis is false, i.e., the algorithm does not stop in a finite number of iterations.

First, we consider the case in which $\alpha_u^t = \infty$ for all t, i.e., the interval upper bound is never updated. In this case, by the instructions of Algorithm 5.3, the Wolfe sufficient decrease condition is always satisfied, i.e., for each step size α^t we have that

$$F\left(x_k + \alpha^t d_k\right) \leq F(x_k) + \mathbf{1}_m \gamma \alpha^t D\left(x_k, d_k\right). \tag{C.12}$$

By hypothesis 2.a., for all t, if $\alpha_u^t = \infty$, then α^t is updated according to (C.11). Using the latter, we obtain that

$$\alpha^t \geq \eta \max\left\{\alpha_l^t, \alpha^0\right\} \geq \eta \alpha_l^t. \tag{C.13}$$

Moreover, Line 10 is executed at every iteration, i.e., $\alpha_l^t = \alpha^{t-1}$. Then, (C.13) turns into $\alpha^t \geq \eta \alpha^{t-1} \geq \eta^2 \max\left\{\alpha_l^{t-1}, \alpha^0\right\} \geq \ldots \geq (\eta)^t \alpha^0$. Since $\eta > 1$ and $\alpha^0 = 1$, it follows that $\lim_{t\to\infty} \alpha^t = \infty$.

Then, there exists an infinite sequence of points $\{x_k + \alpha d_k\}_{\alpha \geq \alpha^0}$ satisfying (C.12). This fact is in contradiction with Assumption 3.1.

Now, we consider the case in which $\exists\, \tilde{t}$ such that $\alpha_u^{\tilde{t}} \leq M$. Then, the bounded and monotone sequences $\{\alpha_l^t\}_{t\geq\tilde{t}}$, with $\alpha_l^t \geq 0$, and $\{\alpha_u^t\}_{t\geq\tilde{t}}$, with $\alpha_u^t \leq M$, are generated. From Lemma 5.1, it follows that, for $t \geq \tilde{t}$,

$$\exists\, j\left(\alpha_u^t\right) \text{ s.t. } f_{j(\alpha_u^t)}\left(x_k + \alpha_u^t d_k\right) > f_{j(\alpha_u^t)}\left(x_k\right) + \gamma \alpha_u^t D\left(x_k, d_k\right), \tag{C.14}$$

$$\forall j \in \{1, \ldots, m\},\ f_j\left(x_k + \alpha_l^t d_k\right) \leq f_j\left(x_k\right) + \gamma \alpha_l^t D\left(x_k, d_k\right). \tag{C.15}$$

Let $\mathcal{T} \subseteq \{\tilde{t}, \tilde{t}+1, \ldots\}$ be a subsequence such that, for all $t \in \mathcal{T}$, $j\left(\alpha_u^t\right) = \hat{j}$.

By the instructions of the algorithm, the upper and lower bounds are updated in one of the following ways: $\alpha_u^{t+1} = \alpha^t$, $\alpha_l^{t+1} = \alpha_l^t$; $\alpha_u^{t+1} = \alpha_u^t$, $\alpha_l^{t+1} = \alpha^t$. In the first case, we can state that

$$\begin{aligned} \max\left\{\left(\alpha^t - \alpha_l^t\right), \left(\alpha_u^t - \alpha^t\right)\right\} &= \max\left\{\left(\alpha_u^{t+1} - \alpha_l^{t+1}\right), \left(\alpha_u^t - \alpha_u^{t+1}\right)\right\} \\ &\geq \alpha_u^{t+1} - \alpha_l^{t+1}. \end{aligned} \tag{C.16}$$

An analogous result can be also achieved for the second case. Using (C.16) and hypothesis 2.b, we obtain that $\alpha_u^{t+1} - \alpha_l^{t+1} \leq \max\left\{\left(\alpha^t - \alpha_l^t\right), \left(\alpha_u^t - \alpha^t\right)\right\} \leq \delta\left(\alpha_u^t - \alpha_l^t\right)$. Recalling that $\delta \in [1/2, 1)$, $\{\alpha_l^t\}$ and $\{\alpha_u^t\}$ are monotone and bounded sequences and that, for all t, we have $\alpha_l^t \leq \alpha^t \leq \alpha_u^t$, the above equation implies that $\alpha_u^t - \alpha_l^t$ goes to zero as $t \to \infty$, with $t \in \mathcal{T}$. Moreover, it follows that

$$\lim_{\substack{t\to\infty \\ t\in\mathcal{T}}} \alpha_u^t = \lim_{\substack{t\to\infty \\ t\in\mathcal{T}}} \alpha_l^t = \lim_{\substack{t\to\infty \\ t\in\mathcal{T}}} \alpha^t = \bar{\alpha}. \tag{C.17}$$

Given (C.14) and (C.15), the definition of $\hat{j}$ and the continuity of $F(\cdot)$, by taking the limit for $t \to \infty$, with $t \in \mathcal{T}$, we obtain that

$$f_{\hat{j}}\left(x_k + \bar{\alpha} d_k\right) = f_{\hat{j}}\left(x_k\right) + \gamma \bar{\alpha} D\left(x_k, d_k\right). \tag{C.18}$$

Taking into account this result and (C.14), we have that, for all $t \in \mathcal{T}$, $\alpha_u^t > \bar{\alpha}$.

On the other hand, for $t \in \mathcal{T}$, Equation (C.14) can be re-written in the following way: $f_{\hat{j}}\left(x_k + \alpha_u^t d_k\right) > f_{\hat{j}}\left(x_k\right) + \gamma\left(\bar{\alpha} + \alpha_u^t - \bar{\alpha}\right) D\left(x_k, d_k\right)$. Using (C.18) and by simple algebraic manipulations we get $f_{\hat{j}}\left(x_k + \alpha_u^t d_k\right) > f_{\hat{j}}\left(x_k + \bar{\alpha} d_k\right) + \gamma\left(\alpha_u^t - \bar{\alpha}\right) D\left(x_k, d_k\right)$ and, then,

$$\frac{f_{\hat{j}}\left(x_k + \alpha_u^t d_k\right) - f_{\hat{j}}\left(x_k + \bar{\alpha} d_k\right)}{\alpha_u^t - \bar{\alpha}} > \gamma D\left(x_k, d_k\right).$$

Now, by taking the limit for $t \to \infty$, with $t \in \mathcal{T}$, and recalling (C.17) and the continuous differentiability of $F(\cdot)$, we obtain that

$$\nabla f_{\hat{j}}\left(x_k + \bar{\alpha} d_k\right)^\top d_k \geq \gamma D\left(x_k, d_k\right). \tag{C.19}$$

Since $D\left(x_k + \bar{\alpha} d_k, d_k\right) \geq \nabla f_{\hat{j}}\left(x_k + \bar{\alpha} d_k\right)^\top d_k$ by definition of $D(\cdot,\cdot)$, $\sigma > \gamma$ and $D\left(x_k, d_k\right) < 0$, from (C.19) we have that

$$D\left(x_k + \bar{\alpha} d_k, d_k\right) > \sigma D\left(x_k, d_k\right). \tag{C.20}$$

However, from Lemma 5.1, it follows that $D\left(x_k + \alpha_l^t d_k, d_k\right) < \sigma D\left(x_k, d_k\right)$. By taking the limit for $t \to \infty$, with $t \in \mathcal{T}$, and recalling (C.17) and the continuity of $D(\cdot,\cdot)$, we get from the last equation that $D\left(x_k + \bar{\alpha} d_k, d_k\right) \leq \sigma D\left(x_k, d_k\right)$. The latter is in contradiction with (C.20). We thus get the thesis. □

Lemma 5.4. *Let Assumptions 5.1 and 5.2 hold. Moreover, let $\{x_k\}$ be the sequence generated by Algorithm 5.4. Then, there exists a constant $\delta > 0$ such that, for all $k \geq 0$, we have that*

$$\cos \beta^k \geq \delta.$$

Proof. Let us consider $k \geq 0$, $\tau \in [0,1]$ and the point $x_k + \tau s_k$. By Assumption 5.1 and Equations (5.3) and (5.5), we have that $x_k + \tau s_k \in \mathcal{L}_F\left(F\left(x_0\right)\right)$. Also recalling that $\lambda^{LM}\left(x_k\right)$ satisfies (2.6), we obtain for any $z \in \mathbb{R}^n$ that

$$\int_0^1 a \left\|z\right\|^2 d\tau \leq \int_0^1 z^\top \sum_{j=1}^m \lambda_j^{LM}\left(x_k\right) \nabla^2 f_j(x_k + \tau s_k) z \, d\tau \leq \int_0^1 b \left\|z\right\|^2 d\tau$$

and, then,

$$a \left\|z\right\|^2 \leq z^\top \int_0^1 \sum_{j=1}^m \lambda_j^{LM}\left(x_k\right) \nabla^2 f_j(x_k + \tau s_k) z \, d\tau \leq b \left\|z\right\|^2. \tag{C.21}$$

For $z = s_k$ we thus obtain

$$a \left\| s_k \right\|^2 \leq s_k^\top \int_0^1 \sum_{j=1}^{m} \lambda_j^{LM} \left(x_k \right) \nabla^2 f_j (x_k + \tau s_k) s_k \, d\tau \leq b \left\| s_k \right\|^2 . \tag{C.22}$$

Defining

$$G_k = \int_0^1 \sum_{j=1}^{m} \lambda_j^{LM} \left(x_k \right) \nabla^2 f_j (x_k + \tau s_k) \, d\tau \tag{C.23}$$

and, recalling (5.6), we solve the integral:

$$\begin{aligned} G_k s_k &= \sum_{j=1}^{m} \lambda_j^{LM} \left(x_k \right) \int_0^1 \nabla^2 f_j (x_k + \tau s_k) s_k \, d\tau \\ &= \sum_{j=1}^{m} \lambda_j^{LM} \left(x_k \right) \left[\nabla f_j \left(x_{k+1} \right) - \nabla f_j \left(x_k \right) \right] = u_k. \end{aligned} \tag{C.24}$$

Given this last result and Equation (C.22), we obtain that $a \left\| s_k \right\|^2 \leq s_k^\top u_k \leq b \left\| s_k \right\|^2$ and, thus, considering the left-hand side,

$$\frac{s_k^\top u_k}{\left\| s_k \right\|^2} \geq a. \tag{C.25}$$

Furthermore, if we consider $z = G_k^{1/2} s_k$ in (C.21), with $G_k^{1/2}$ being the positive definite square root of G_k, we get

$$a \left\| G_k^{1/2} s_k \right\|^2 \leq \left(G_k^{1/2} s_k \right)^\top \int_0^1 \sum_{j=1}^{m} \lambda_j^{LM} \left(x_k \right) \nabla^2 f_j (x_k + \tau s_k) \, d\tau \left(G_k^{1/2} s_k \right) \leq b \left\| G_k^{1/2} s_k \right\|^2$$

and, recalling (C.23), $a \left(s_k^\top G_k s_k \right) \leq s_k^\top G_k^2 s_k \leq b \left(s_k^\top G_k s_k \right)$. Then, given Remark 5.5 and Equation (C.24), focusing on the right-hand side, we have

$$\frac{\left\| u_k \right\|^2}{s_k^\top u_k} \leq b.$$

Now, recalling Assumption 5.2 and Equation (5.10), we apply recursively (5.11) and we obtain that

$$\begin{aligned} \mathrm{Tr}(B^{k+1}) &= \mathrm{Tr}(B_{(0)}^k) - \sum_{l=0}^{\min\{k, M-1\}} \frac{\left\| B_{(l)}^k s_{l+h} \right\|^2}{s_{l+h}^\top B_{(l)}^k s_{l+h}} + \sum_{l=0}^{\min\{k, M-1\}} \frac{\left\| u_{l+h} \right\|^2}{s_{l+h}^\top u_{l+h}} \\ &\leq \mathrm{Tr}(B^0) + \sum_{l=0}^{\min\{k, M-1\}} \frac{\left\| u_{l+h} \right\|^2}{s_{l+h}^\top u_{l+h}} \\ &\leq \mathrm{Tr}(B^0) + \left(\min\left\{ k, M-1 \right\} + 1 \right) b \leq \tilde{b}, \end{aligned} \tag{C.26}$$

for some $\tilde{b} > 0$, where the inequalities come from the fact that, for all $k \geq 0$ and $l = 0, \ldots, \min\{k, M-1\}$, $B^k_{(l)}$ is positive definite (cf. the instructions of Algorithm 5.4 and Remark 5.5). We can apply a similar reasoning with the determinant formula (5.12):

$$\begin{aligned}\det(B^{k+1}) &= \det(B^k_{(0)}) \prod_{l=0}^{\min\{k,M-1\}} \frac{s_{l+h}^\top u_{l+h}}{s_{l+h}^\top B^k_{(l)} s_{l+h}} \\ &= \det(B^0) \prod_{l=0}^{\min\{k,M-1\}} \frac{s_{l+h}^\top u_{l+h}}{\|s_{l+h}\|^2} \frac{\|s_{l+h}\|^2}{s_{l+h}^\top B^k_{(l)} s_{l+h}}.\end{aligned}$$

From (C.26), we deduce that the greatest eigenvalue of $B^k_{(l)}$ is smaller than $\tilde{b}$. Thus, given Assumption 5.2 and Equation (C.25), we get that

$$\det\left(B^{k+1}\right) \geq \det(B^0) \left(\frac{a}{\tilde{b}}\right)^{\min\{k,M-1\}+1} \geq \tilde{a}, \tag{C.27}$$

where $\tilde{a} > 0$.

Then, by (5.13), the min-max theorem and the triangle inequality, we have:

$$\cos\beta^k = \frac{s_k^\top B^k s_k}{\|s_k\| \|B^k s_k\|} \geq \frac{\omega_m\left(B^k\right)\|s_k\|^2}{\|B^k\|\|s_k\|^2} = \frac{\omega_m\left(B^k\right)}{\|B^k\|}.$$

We know that:

- by definition of trace and determinant, recalling (C.26) and (C.27), we get
$$\det\left(B^k\right) = \prod_{\omega \in \Omega(B^k)} \omega \leq (n-1)\omega_M\left(B^k\right)\omega_m\left(B^k\right),$$
and thus
$$\omega_m\left(B^k\right) \geq \frac{\det\left(B^k\right)}{(n-1)\,\omega_M\left(B^k\right)} \geq \frac{\tilde{a}}{(n-1)\,\mathrm{Tr}\left(B^k\right)} \geq \frac{\tilde{a}}{(n-1)\,\tilde{b}};$$
- considering the euclidean norm and that B^k is a real positive definite matrix, $\left\|B^k\right\| \leq \omega_M\left(B^k\right) \leq \mathrm{Tr}\left(B^k\right) \leq \tilde{b}$.

Joining the last three results, we obtain that

$$\cos\beta^k \geq \frac{\omega_m\left(B^k\right)}{\|B^k\|} \geq \frac{\tilde{a}}{(n-1)\,\tilde{b}^2} > 0,$$

where the last inequality comes from the definitions of $\tilde{a}$ and $\tilde{b}$. Thus, we get the thesis choosing $\delta = \frac{\tilde{a}}{(n-1)\tilde{b}^2}$. □

Proposition 6.3. *If X^k contains mutually nondominated points with respect to $F(\cdot)$, then $\hat{X}^k$ contains nondominated points at any time during iteration k; thus Step 15 of Algorithm 6.1 is always well defined and X^{k+1} is finally a set of nondominated solutions.*

Proof. At iteration k, the set $\hat{X}^k$ is initialized with the nondominated points X^k; then, it is only updated at Steps 12 and 16. At Step 12, either $z_c^k = x_c$, and the set is not modified, or, by the definition of α_c^k, z_c^k dominates x_c, which in turn was nondominated. Thus, the added point z_c^k is nondominated, while all the newly dominated points are removed. At Step 16, the added point $z_c^k + \alpha_c^{\mathcal{I}} v_{\mathcal{I}}^{SD}(z_c^k)$ is nondominated by the definition of $\alpha_c^{\mathcal{I}}$; all the newly dominated points are removed. Thus, $\hat{X}^k$ always contains mutually nondominated solutions. By Proposition 6.2, Step 15 is therefore always well defined. Moreover, since $X^{k+1} = \hat{X}^k$ at the end of the iteration, X^{k+1} inherits the nondominance property from $\hat{X}^k$. □

Lemma 6.1. *After Step 12 of Algorithm 6.1, z_c^k belongs to $\hat{X}^k$. Moreover, for all $\tilde{k} > k$, there exists $y \in X^{\tilde{k}}$ such that $F(y) \leq F(z_c^k)$.*

Proof. The first assertion of the proposition trivially follows from the update rule of $\hat{X}^k$, at Step 12. Now, either $z_c^k \in X^{\tilde{k}}$ or $z_c^k \notin X^{\tilde{k}}$; in the former case, we trivially have $y = z_c^k$; otherwise, we can notice that, by the instructions of the algorithm, any set $X^{\tilde{k}}$, $\tilde{k} > k$, is the result of repeated application of Steps 12 and 16, starting from $\hat{X}^k$ at some point when $z_c^k \in \hat{X}^k$. When z_c^k was removed from the set, a point y^1 was certainly inserted such that $F(y^1) \leq F(z_c^k)$. Then, either $y^1 \in X^{\tilde{k}}$, or y^1 was removed when a point y^2 such that $F(y^2) \leq F(y^1)$ was added. By recursively applying the reasoning, we have that there is certainly a point $y^t \in X^{\tilde{k}}$ such that $F(y^t) \leq F(y^{t-1}) \leq \ldots \leq F(y^2) \leq F(y^1) \leq F(z_c^k)$. This completes the proof. □

Proposition 7.1. *Let $\{X^{k+1}\}$ be the sequence of sets generated by Algorithm 7.1. Then, for each k and for each $x_{k+1} \in X^{k+1}$, we have:*

1. *x_{k+1} is not dominated by any other point in X^{k+1} w.r.t. $\mathcal{L}^{\tau_k}(\cdot, \mu^k)$, i.e., there does not exist $y \in X^{k+1}$ such that $\mathcal{L}^{\tau_k}(y, \mu^k) \lneqq \mathcal{L}^{\tau_k}(x_{k+1}, \mu^k)$;*
2. *x_{k+1} is ε_k-Pareto-stationary w.r.t. $\mathcal{L}^{\tau_k}(\cdot, \mu^k)$.*

Proof. We prove the two statements one at a time:

1. X^{k+1} is equal to X_{tmp} at the end of the main loop of each iteration, at Line 17. X_{tmp} is initialized with $\hat{X}^k$, which contains mutually nondominated points w.r.t. $\mathcal{L}^{\tau_k}(\cdot, \mu^k)$ by its definition at Line 4. Then, X_{tmp} can be modified only at Lines 11-16. In the first case, we have that the point $z_c^{\mathcal{I}}$, with $\mathcal{I} \subseteq \{1, \ldots, m\}$, is non-dominated by the definition of $\alpha_c^{\mathcal{I}}$ and the `MOSD` procedure; in the second case, by the condition at Line 15, a point is added only if it is non-dominated. In both scenarios, if the new point is added in X_{tmp}, all the solutions dominated by it are removed from the set.

2. We have two possible cases: $x_{k+1} \in \hat{X}^k$ or $x_{k+1} \notin \hat{X}^k$. In the latter case, x_{k+1} has necessarily been added to X_{tmp} through instructions 11-16; in particular, x_{k+1} was produced by either instruction 10 or 14 and is thus ε_k-Pareto-stationary.

   So, let us assume that $x_{k+1} \in \hat{X}^k$ and, by contradiction, that $\theta_k^{SD}(x_{k+1}) < -\varepsilon_k$. In this case, $x_c = x_{k+1}$ would satisfy the condition at Step 12. The line search hence is guaranteed, by Proposition 3.2 ($\Omega = \mathbb{R}^n$), to find a step α_c such that $\mathcal{L}^{\tau_k}(x_c + \alpha_c v_k^{SD}(x_c), \mu^k) < \mathcal{L}^{\tau_k}(x_c, \mu^k)$, and by the properties of the `MOSD` procedure we have $\mathcal{L}^{\tau_k}(z_c, \mu^k) \leq \mathcal{L}^{\tau_k}(x_c + \alpha_c v_k^{SD}(x_c), \mu^k)$. Hence, this new point z_c (strictly) dominates x_{k+1} w.r.t. $\mathcal{L}^{\tau_k}(\cdot, \mu^k)$. Now, from the instructions of the algorithm, either z_c belongs to X^{k+1} or there exists $y \in X^{k+1}$ such that $\mathcal{L}^{\tau_k}(y, \mu^k) \lneqq \mathcal{L}^{\tau_k}(z_c, \mu^k) < \mathcal{L}^{\tau_k}(x_{k+1}, \mu^k)$. However, this is absurd, since $x_{k+1} \in X^{k+1}$ and from statement 1. X^{k+1} contains mutually nondominated points. Hence, $\theta_k^{SD}(x_{k+1}) \geq -\varepsilon_k$.

□

Lemma 7.1. *Let $\{X^k\}$ be the sequence of sets generated by Algorithm 7.1, and let $\{x_k\}$ be any sequence of points such that $x_k \in X^k$ for all k. Let $\bar{x}$ be a limit point of $\{x_k\}$, i.e., there exists an infinite subset $K \subseteq \{0, 1, \ldots\}$ such that $\lim_{k \to \infty, k \in K} x_k = \bar{x}$, and suppose that $g(\bar{x}) \leq \mathbf{0}_p$, i.e., $\bar{x} \in \Omega$. Then, for all $i = 1, \ldots, p$ such that $g_i(\bar{x}) < 0$ we have*

$$\max\{0, \mu_i^k + \tau_k g_i(x_{k+1})\} = 0$$

for all $k \in K$ sufficiently large.

Proof. Let $g_i(\bar{x}) < 0$ and $k_1 \in K$ be such that $g_i(x_{k+1}) < c < 0$ for all $k \geq k_1$, $k \in K$. From the instructions of the algorithm we know that $\mu_i^k \geq 0$ for all k. There are two possible cases.

- $\tau_k \to \infty$
  The sequence $\{\mu^k\}$ is bounded by definition, hence there exists $k_2 \geq k_1$, $k_2 \in K$, such that for all $k \in K$, $k \geq k_2$ we have $\mu_i^k + \tau_k g_i(x_{k+1}) < 0$ and thus $\max\{0, \mu_i^k + \tau_k g_i(x_{k+1})\} = 0$.

- $\{\tau_k\}$ is bounded.
  From instruction 21 of the algorithm, there must exist $k_2 \geq k_1$ such that, for all $k \geq k_2$, condition

  $$\forall\, x \in X^{k+1} \quad \mu_j^k + \tau_k g_j(x) \leq 0 \quad \forall j \in \{1, \ldots, p\} \text{ s.t. } g_j(x) < 0$$

  holds. Hence, for $k \geq k_2$, $k \in K$, we have $\mu_i^k + \tau_k g_i(x_{k+1}) \leq 0$. Thus, we have $\max\{0, \mu_i^k + \tau_k g_i(x_{k+1})\} = 0$ for $k \in K$ sufficiently large.

□

Lemma 8.2. *Assume $F(\cdot)$ is component-wise convex. Let $\bar{x} \in \Omega$ a Pareto-stationary point for Problem* (8.1)*. Then, $\bar{x}$ is locally weakly Pareto optimal for* (8.1)*.*

Proof. Since $\bar{x}$ is a Pareto-stationarity point for Problem (8.1), by Definition 8.1, we have that $\theta^S(\bar{x}) = 0$, i.e.,

$$\max_{j \in \{1,\ldots,m\}} \nabla f_j\left(\bar{x}\right)^\top d + \frac{1}{2}\left\|d\right\|^2 \geq 0, \qquad \forall d \in \mathcal{D}\left(\bar{x}\right). \tag{C.28}$$

Let us suppose, by contradiction, that there exists a direction $\hat{d} \in \mathcal{D}(\bar{x})$ such that

$$\max_{j \in \{1,\ldots,m\}} \nabla f_j\left(\bar{x}\right)^\top \hat{d} < 0. \tag{C.29}$$

Since (C.28) holds, we then deduce that $\frac{1}{2}\|\hat{d}\|^2 \geq |\max_{j\in\{1,\ldots,m\}} \nabla f_j(\bar{x})^\top \hat{d}\,| > 0$. Now, let us introduce the function $\tilde{\theta}^S : \mathbb{R}^n \times \mathbb{R}^n \times [0,1] \to \mathbb{R}$ as $\tilde{\theta}^S(x,d,t) = \max_{j\in\{1,\ldots,m\}} \nabla f_j(x)^\top (td) + \frac{1}{2}\|td\|^2 = t \max_{j\in\{1,\ldots,m\}} \nabla f_j(x)^\top d + \frac{t^2}{2}\|d\|^2$. By (8.4) and the feasibility of $\hat{d}$, it follows that, for all $t \in [0,1]$, $t\hat{d} \in \mathcal{D}(\bar{x})$ and $\theta^S(\bar{x}) \leq \tilde{\theta}^S(\bar{x}, \hat{d}, t)$. It is easy to see that $\tilde{\theta}^S(\bar{x}, \hat{d}, t) < 0$ if $0 < t < -(2/\|\hat{d}\|^2) \max_{j\in\{1,\ldots,m\}} \nabla f_j(\bar{x})^\top \hat{d}$, where the right-hand side is

a positive quantity by Equation (C.29). But, in this case, we would have that $\theta^S(\bar{x}) \leq \tilde{\theta}^S(\bar{x}, \hat{d}, t) < 0$, which contradicts the Pareto-stationarity of $\bar{x}$. Thus, we prove that, if $\bar{x}$ is Pareto stationary for Problem (8.1), then $\max_{j\in\{1,\dots,m\}} \nabla f_j(\bar{x})^\top d \geq 0,\ \forall d \in \mathcal{D}(\bar{x})$. From this point forward, we can follow the proof of Proposition 3.5 in [57]. □

Lemma 8.7. *Let Assumptions 3.1-8.1 hold and $\{x_k\}$ be the sequence generated by Algorithm 8.1 with constant $L > \max_{j\in\{1,\dots,m\}} L(f_j)$. Then:*

1. *for all k, $F(x_k) - F(x_{k+1}) \geq \frac{1}{2}\|x_k - x_{k+1}\|^2(\mathbf{1}_m L - L(F))$;*
2. *for all k, if $x_k \neq x_{k+1}$, then $F(x_{k+1}) < F(x_k)$;*
3. *for all $j \in \{1,\dots,m\}$, the sequence $\{f_j(x_k)\}$ is non-increasing;*
4. *the sequence $\{F(x_k)\}$ converges;*
5. *$\lim_{k\to\infty} \|x_k - x_{k+1}\|^2 = 0$.*

Proof.

1. The thesis can be proved making an argument similar to that of Proposition 8.2 and reminding that $x_{k+1} = x_k + v_k^L$, with $v_k^L \in \mathcal{V}^L(x_k)$ (Step 6 of Algorithm 8.1).
2. It follows directly from Point 1., recalling that $L > L(f_j)$ for all $j \in \{1,\dots,m\}$ and $\|x_k - x_{k+1}\| > 0$.
3. By Point 1. and the hypothesis on L, we have that, for all k and $j \in \{1,\dots,m\}$, $f_j(x_{k+1}) \leq f_j(x_k)$. Thus, for all $j \in \{1,\dots,m\}$, the sequence $\{f_j(x_k)\}$ is non-increasing.
4. It follows from Assumption 3.1 and Point 3..
5. From Point 1., we have that, for all k and $j \in \{1,\dots,m\}$, $\frac{L-L(f_j)}{2}\|x_k - x_{k+1}\|^2 \leq f_j(x_k) - f_j(x_{k+1})$. Since $f_j(\cdot)$ is continuous, we can take the limit for $k \to \infty$ on both sides of the inequality: $\lim_{k\to\infty} \frac{L-L(f_j)}{2}\|x_k - x_{k+1}\|^2 \leq \lim_{k\to\infty} f_j(x_k) - f_j(x_{k+1}) = 0$, where the equality comes from Point 4.. By the definition of L and the non-negativity of the norm, the statement is proved.

□

Appendix D

Novel and Modified Test Problems

In this appendix, we introduce two new convex MOO test problems, which we call MAN_1 and MAN_2. Moreover, we report the formulation of rescaled versions of FDS_1 [34] and MOP_2 [50]. Finally, we introduce a modified version of the OSY problem [85] with convex feasible set and convex objective functions. All the mentioned problems can be also found listed in Table 9.1.

MAN_1

$$\begin{aligned} & & f_1(x) &= \textstyle\sum_{i=1}^{n} (x_i - i)^2 \\ \min & & f_2(x) &= \textstyle\sum_{i=1}^{n} e^{-x_i} + x_i \\ & \text{s.t.} \quad x \in & \left[-10^4, 10^4\right]^n . & \end{aligned}$$

MAN_2

$$\begin{aligned} & & f_1(x) &= \textstyle\sum_{i=1}^{n} \frac{i(x_i - i)^2}{n^2} \\ \min & & f_2(x) &= \textstyle\sum_{i=1}^{n} e^{-x_i} + x_i \\ & & f_3(x) &= \textstyle\sum_{i=1}^{n} e^{x_i^2} \\ & \text{s.t.} \quad x \in & [-1, 1]^n . & \end{aligned}$$

M-FDS_1

$$\begin{aligned} & f_1(x) = \sum_{i=1}^{n} \frac{i(x_i - i)^4}{n^4} \\ \min \quad & f_2(x) = e^{\sum_{i=1}^{n} x_i/n} + \|x\|^2 \\ & f_3(x) = \sum_{i=1}^{n} \frac{i(n-i+1)e^{-x_i}}{n(n+1)} \\ \text{s.t.} \quad & x \in [-2, 2]^n . \end{aligned}$$

M-MOP_2

$$\begin{aligned} & f_1(x) = 1 - e^{-\sum_{i=1}^{n}(x_i - 1/\sqrt{n})^2/n} \\ \min \quad & \\ & f_2(x) = 1 - e^{-\sum_{i=1}^{n}(x_i + 1/\sqrt{n})^2/n} \\ \text{s.t.} \quad & x \in [-4, 4]^n . \end{aligned}$$

M-OSY

$$\begin{aligned} & f_1(x) = 25(x_1 - 2)^2 + (x_2 - 2)^2 + (x_3 - 1)^2 + (x_4 - 4)^2 + (x_5 - 1)^2 \\ \min_{x \in \mathbb{R}^6} \quad & \\ & f_2(x) = \sum_{i=1}^{6} x_i^2 \end{aligned}$$

$$\begin{aligned} \text{s.t.} \quad & x_1 + x_2 - 2 \geq 0, \\ & 6 - x_1 - x_2 \geq 0, \\ & 2 - x_2 + x_1 \geq 0, \\ & 2 - x_1 + 3x_2 \geq 0, \\ & 4 - (x_3 - 3)^2 - x_4 \geq 0, \\ & -(x_5 - 3)^2 + x_6 - 4 \geq 0, \\ & 0 \leq x_1, x_2, x_6 \leq 10, \\ & 1 \leq x_3, x_5 \leq 5, \\ & 0 \leq x_4 \leq 6. \end{aligned}$$

Appendix E

Publications

This research activity has led to several publications in international journals [1]. For some of them, a public version of the experimental code has been also published.

International Journals

1. **P. Mansueto**, F. Schoen "Memetic differential evolution methods for clustering problems", *Pattern Recognition*, 114 (2021) [DOI: 10.1016/j.patcog.2021.107849]
2. G. Cocchi, M. Lapucci, **P. Mansueto** "Pareto front approximation through a multi-objective augmented Lagrangian method", *EURO Journal on Computational Optimization*, 9 (2021) [DOI: 10.1016/j.ejco.2021.100008]
3. M. Lapucci, **P. Mansueto**, F. Schoen, "A memetic procedure for global multi-objective optimization", *Mathematical Programming Computation*, 15, 227–267 (2023) [DOI: 10.1007/s12532-022-00231-3]
4. M. Lapucci, **P. Mansueto**, "A limited memory Quasi-Newton approach for multi-objective optimization", *Computational Optimization and Applications*, 85, 33–73 (2023) [DOI: 10.1007/s10589-023-00454-7]
5. M. Lapucci, **P. Mansueto**, "Improved front steepest descent for multi-objective optimization", *Operations Research Letters*, 51, 242–247 (2023) [DOI: 10.1016/j.orl.2023.03.001]
6. M. Lapucci, **P. Mansueto**, "Cardinality-Constrained Multi-Objective Optimization: Novel Optimality Conditions and Algorithms", Accepted at *Jour-*

[1] The author's bibliometric indices are the following: H-index = 4, total number of citations = 44 (source: Google Scholar on January 29th, 2024).

nal of Optimization Theory and Applications (2024; Available soon) [Preprint DOI: 10.48550/arXiv.2304.02369]

7. S. Marinai, S. Capobianco, Z. Ziran, A. Giuntini, **P. Mansueto**, "Recognition of Concordances for Indexing in Digital Libraries", In: *Ceci, M., Ferilli, S., Poggi, A. (eds) Digital Libraries: The Era of Big Data and Data Science, IRCDL 2020, Communications in Computer and Information Science*, 1177 (2020) [DOI: 10.1007/978-3-030-39905-4_14]

Experimental Codes

1. **P. Mansueto**, "A Memetic Procedure for Global Multi-Objective Optimization" (2022) [DOI: 10.5281/zenodo.7299857] [GitHub Repository URL: `https://github.com/pierlumanzu/nsma`]
2. **P. Mansueto**, "LM-Q-NWT: A Limited Memory Quasi-Newton Approach for Multi-Objective Optimization" (2023) [DOI: 10.5281/zenodo.7762660] [GitHub Repository URL: `https://github.com/pierlumanzu/limited_memory_method_for_MOO`]
3. **P. Mansueto**, "Improved Front Steepest Descent for Multi-Objective Optimization" (2023) [DOI: 10.5281/zenodo.7762661] [GitHub Repository URL: `https://github.com/pierlumanzu/ifsd`]
4. **P. Mansueto**, "Pareto front approximation through a multi-objective augmented Lagrangian method" (2023) [DOI: 10.5281/zenodo.8337581] [GitHub Repository URL: `https://github.com/pierlumanzu/front-alamo`]
5. **P. Mansueto**, M. Lapucci, "MOIHT & SFSD – Algorithms for Cardinality-Constrained Multi-Objective Optimization Problems" (2024) [DOI: 10.5281/zenodo.10473331] [GitHub Repository URL: `https://github.com/pierlumanzu/cc-moo`]

Bibliography

[1] M. Ansary and G. Panda, "A modified Quasi-Newton method for vector optimization problem," *Optimization*, vol. 64, no. 11, pp. 2289–2306, 2015.

[2] S. Bandyopadhyay, S. Saha, U. Maulik, and K. Deb, "A simulated annealing-based multiobjective optimization algorithm: AMOSA," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 3, pp. 269–283, 2008.

[3] A. Beck and Y. C. Eldar, "Sparsity constrained nonlinear optimization: Optimality conditions and algorithms," *SIAM Journal on Optimization*, vol. 23, no. 3, pp. 1480–1509, 2013.

[4] A. Beck and N. Hallak, "On the minimization over sparse symmetric sets: Projections, optimality conditions, and algorithms," *Mathematics of Operations Research*, vol. 41, no. 1, pp. 196–223, 2016.

[5] D. P. Bertsekas, *Nonlinear programming*, 2nd ed. Belmont, MA: Athena Scientific, 1999.

[6] D. Bertsimas, A. Delarue, P. Jaillet, and S. Martin, "The price of interpretability," *arXiv preprint arXiv:1907.03419*, 2019.

[7] D. Bertsimas and A. King, "Logistic regression: From art to science," *Statistical Science*, vol. 32, no. 3, pp. 367–384, 2017.

[8] D. Bertsimas, A. King, and R. Mazumder, "Best subset selection via a modern optimization lens," *The Annals of Statistics*, vol. 44, no. 2, pp. 813 – 852, 2016.

[9] D. Bertsimas, J. Pauphilet, and B. Van Parys, "Sparse regression: Scalable algorithms and empirical performance," *Statistical Science*, vol. 35, no. 4, pp. 555–578, 2020.

[10] J. Bhuvana and C. Aravindan, "Memetic algorithm with preferential local search using adaptive weights for multi-objective optimization problems," *Soft Computing*, vol. 20, 02 2015.

[11] D. Bienstock, "Computational study of a family of mixed-integer quadratic programming problems," *Mathematical Programming*, vol. 74, no. 2, pp. 121–140, Aug 1996.

[12] E. G. Birgin and J. M. Martinez, *Practical augmented Lagrangian methods for constrained optimization.* SIAM, 2014, vol. 10.

[13] H. Bonnel, A. N. Iusem, and B. F. Svaiter, "Proximal methods in vector optimization," *SIAM Journal on Optimization*, vol. 15, no. 4, pp. 953–970, 2005.

[14] M. Brown and R. E. Smith, "Directed multi-objective optimization," *International Journal of Computers, Systems, and Signals*, vol. 6, no. 1, pp. 3–17, 2005.

[15] O. P. Burdakov, C. Kanzow, and A. Schwartz, "Mathematical programs with cardinality constraints: reformulation by complementarity-type conditions and a regularization method," *SIAM Journal on Optimization*, vol. 26, no. 1, pp. 397–425, 2016.

[16] F. Cabassi and M. Locatelli, "Computational investigation of simple memetic approaches for continuous global optimization," *Computers & Operations Research*, vol. 72, pp. 50–70, 2016.

[17] E. F. Campana, M. Diez, G. Liuzzi, S. Lucidi, R. Pellegrini, V. Piccialli, F. Rinaldi, and A. Serani, "A multi-objective direct algorithm for ship hull optimization," *Computational Optimization and Applications*, vol. 71, no. 1, pp. 53–72, 2018.

[18] M. A. Carreira-Perpiñán and Y. Idelbayev, ""Learning-compression" algorithms for neural net pruning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[19] E. Carrizosa and J. B. G. Frenk, "Dominating sets for convex functions with some applications," *Journal of Optimization Theory and Applications*, vol. 96, no. 2, pp. 281–295, 1998.

[20] E. Civitelli, M. Lapucci, F. Schoen, and A. Sortino, "An effective procedure for feature subset selection in logistic regression based on information criteria," *Computational Optimization and Applications*, vol. 80, no. 1, pp. 1–32, Sep 2021.

[21] G. Cocchi and M. Lapucci, "An augmented Lagrangian algorithm for multi-objective optimization," *Computational Optimization and Applications*, vol. 77, no. 1, pp. 29–56, 2020.

[22] G. Cocchi, M. Lapucci, and P. Mansueto, "Pareto front approximation through a multi-objective augmented lagrangian method," *EURO Journal on Computational Optimization*, p. 100008, 2021.

[23] G. Cocchi, G. Liuzzi, S. Lucidi, and M. Sciandrone, "On the convergence of steepest descent methods for multiobjective optimization," *Computational Optimization and Applications*, pp. 1–27, 2020.

[24] G. Cocchi, G. Liuzzi, A. Papini, and M. Sciandrone, "An implicit filtering algorithm for derivative-free multiobjective optimization with box constraints," *Computational Optimization and Applications*, vol. 69, no. 2, pp. 267–296, 2018.

[25] A. L. Custódio, J. A. Madeira, A. I. F. Vaz, and L. N. Vicente, "Direct multisearch for multiobjective optimization," *SIAM Journal on Optimization*, vol. 21, no. 3, pp. 1109–1140, 2011.

[26] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, "A fast and elitist multiobjective genetic algorithm: NSGA-II," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 182–197, 2002.

[27] E. D. Dolan and J. J. Moré, "Benchmarking optimization software with performance profiles," *Mathematical Programming*, vol. 91, no. 2, pp. 201–213, 2002.

[28] M. Drugan and D. Thierens, "Stochastic pareto local search: Pareto neighbourhood exploration and perturbation strategies," *Journal of Heuristics*, vol. 18, 10 2012.

[29] L. G. Drummond and A. N. Iusem, "A projected gradient method for vector optimization problems," *Computational Optimization and applications*, vol. 28, no. 1, pp. 5–29, 2004.

[30] L. G. Drummond, N. Maculan, and B. F. Svaiter, "On the choice of parameters for the weighting method in vector optimization," *Mathematical Programming*, vol. 111, no. 1-2, pp. 201–216, 2008.

[31] D. Dua and C. Graff, "UCI machine learning repository," 2017.

[32] G. Eichfelder, "An adaptive scalarization method in multiobjective optimization," *SIAM Journal on Optimization*, vol. 19, no. 4, pp. 1694–1718, 2009.

[33] E. Filatovas, A. Lančinskas, O. Kurasova, and J. Žilinskas, "A preference-based multi-objective evolutionary algorithm R-NSGA-II with stochastic local search," *Central European Journal of Operations Research*, vol. 25, no. 4, pp. 859–878, 2017.

[34] J. Fliege, L. G. Drummond, and B. F. Svaiter, "Newton's method for multiobjective optimization," *SIAM Journal on Optimization*, vol. 20, no. 2, pp. 602–626, 2009.

[35] J. Fliege, "Gap-free computation of Pareto-points by quadratic scalarizations," *Mathematical Methods of Operations Research*, vol. 59, no. 1, pp. 69–89, 2004.

[36] J. Fliege and B. F. Svaiter, "Steepest descent methods for multicriteria optimization," *Mathematical Methods of Operations Research*, vol. 51, no. 3, pp. 479–494, 2000.

[37] J. Fliege and A. I. F. Vaz, "A method for constrained multiobjective optimization based on SQP techniques," *SIAM Journal on Optimization*, vol. 26, no. 4, pp. 2091–2119, 2016.

[38] E. Fukuda and L. Drummond, "A survey on multiobjective descent methods," *Pesquisa Operacional*, vol. 34, pp. 585–620, 09 2014.

[39] E. H. Fukuda and L. G. Drummond, "On the convergence of the projected gradient method for vector optimization," *Optimization*, vol. 60, no. 8-9, pp. 1009–1021, 2011.

[40] ——, "Inexact projected gradient method for vector optimization," *Computational Optimization and Applications*, vol. 54, no. 3, pp. 473–493, 2013.

[41] R. Garmanjani, E. Krulikovski, and A. Ramos, "On stationarity conditions and constraint qualifications for multiobjective optimization problems with cardinality constraints," 2022.

[42] S. Gass and T. Saaty, "The computational algorithm for the parametric objective function," *Naval Research Logistics Quarterly*, vol. 2, no. 1-2, pp. 39–45, 1955.

[43] A. M. Geoffrion, "Proper efficiency and the theory of vector maximization," *Journal of Mathematical Analysis and Applications*, vol. 22, no. 3, pp. 618–630, 1968.

[44] M. L. N. Gonçalves, F. S. Lima, and L. F. Prudente, "Globally convergent Newton-type methods for multiobjective optimization," *Computational Optimization and Applications*, vol. 83, no. 2, pp. 403–434, Nov 2022.

[45] M. Gravel, J. M. Martel, R. Nadeau, W. Price, and R. Tremblay, "A multicriterion view of optimal resource allocation in job-shop production," *European Journal of Operational Research*, vol. 61, no. 1-2, pp. 230–244, 1992.

[46] D. Gribel and T. Vidal, "Hg-means: A scalable hybrid genetic algorithm for minimum sum-of-squares clustering," *Pattern Recognition*, vol. 88, pp. 569–583, 2019.

[47] A. Grosso, M. Locatelli, and F. Schoen, "A population-based approach for hard global optimization problems based on dissimilarity measures," *Mathematical Programming*, vol. 110, no. 2, pp. 373–404, 2007.

[48] Gurobi Optimization, LLC, "Gurobi Optimizer Reference Manual," 2023. [Online]. Available: https://www.gurobi.com

[49] X. Hu, Z. Huang, and Z. Wang, "Hybridization of the multi-objective evolutionary algorithms and the gradient-based algorithms," in *The 2003 Congress on Evolutionary Computation, 2003. CEC'03.*, vol. 2. IEEE, 2003, pp. 870–877.

[50] S. Huband, P. Hingston, L. Barone, and L. While, "A review of multiobjective test problems and a scalable test problem toolkit," *IEEE Transactions on Evolutionary Computation*, vol. 10, no. 5, pp. 477–506, 2006.

[51] Y. Jin, M. Olhofer, and B. Sendhoff, "Dynamic weighted aggregation for evolutionary multi-objective optimization: Why does it work and how?" in *Proceedings of the genetic and evolutionary computation conference*, 2001, pp. 1042–1049.

[52] Y. Jin and B. Sendhoff, "Pareto-based multiobjective machine learning: An overview and case studies," *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 38, no. 3, pp. 397–415, 2008.

[53] C. Kanzow and M. Lapucci, "Inexact penalty decomposition methods for optimization problems with geometric constraints," *Computational Optimization and Applications*, vol. 85, no. 3, pp. 937–971, Jul 2023.

[54] C. Kanzow and D. Steck, "An example comparing the standard and safeguarded augmented Lagrangian methods," *Operations Research Letters*, vol. 45, no. 6, pp. 598–603, 2017.

[55] H. Kim and M.-S. Liou, "Adaptive directional local search strategy for hybrid evolutionary multiobjective optimization," *Applied Soft Computing Journal*, vol. 19, pp. 290–311, 2014.

[56] A. Konak, D. W. Coit, and A. E. Smith, "Multi-objective optimization using genetic algorithms: A tutorial," *Reliability Engineering & System Safety*, vol. 91, no. 9, pp. 992–1007, 2006.

[57] M. Lapucci, "A penalty decomposition approach for multi-objective cardinality-constrained optimization problems," *Optimization Methods and Software*, vol. 37, no. 6, pp. 2157–2189, 2022.

[58] M. Lapucci, T. Levato, F. Rinaldi, and M. Sciandrone, "A unifying framework for sparsity-constrained optimization," *Journal of Optimization Theory and Applications*, Sep 2023.

[59] M. Lapucci and P. Mansueto, "Improved front steepest descent for multiobjective optimization," *Operations Research Letters*, vol. 51, no. 3, pp. 242–247, 2023.

[60] ——, "A limited memory Quasi-Newton approach for multi-objective optimization," *Computational Optimization and Applications*, vol. 85, no. 1, pp. 33–73, May 2023.

[61] M. Lapucci, P. Mansueto, and F. Schoen, "A memetic procedure for global multi-objective optimization," *Mathematical Programming Computation*, vol. 15, no. 2, pp. 227–267, Jun 2023.

[62] A. Lara, G. Sanchez, C. A. C. Coello, and O. Schutze, "Hcs: A new local search strategy for memetic multiobjective evolutionary algorithms," *IEEE Transactions on Evolutionary Computation*, vol. 14, no. 1, pp. 112–132, 2010.

[63] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler, "Combining convergence and diversity in evolutionary multiobjective optimization," *Evolutionary Computation*, vol. 10, no. 3, pp. 263–282, 2002.

[64] D. Liu, K. C. Tan, C. K. Goh, and W. K. Ho, "A multiobjective memetic algorithm based on particle swarm optimization," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 37, no. 1, pp. 42–50, 2007.

[65] D. C. Liu and J. Nocedal, "On the limited memory BFGS method for large scale optimization," *Mathematical Programming*, vol. 45, no. 1, pp. 503–528, Aug 1989.

[66] T. Liu, X. Gao, and Q. Yuan, "An improved gradient-based NSGA-II algorithm by a new chaotic map model," *Soft Computing*, vol. 21, no. 23, pp. 7235–7249, 2017.

[67] G. Liuzzi, S. Lucidi, and F. Rinaldi, "A derivative-free approach to constrained multiobjective nonsmooth optimization," *SIAM Journal on Optimization*, vol. 26, no. 4, pp. 2744–2774, 2016.

[68] M. Locatelli, M. Maischberger, and F. Schoen, "Differential evolution methods based on local searches," *Computers & Operations Research*, vol. 43, pp. 169–180, 2014.

[69] M. Locatelli and F. Schoen, *Global optimization: theory, algorithms, and applications*. SIAM, 2013.

[70] Z. Lu and Y. Zhang, "Sparse approximation via penalty decomposition methods," *SIAM Journal on Optimization*, vol. 23, no. 4, pp. 2448–2478, 2013.

[71] L. R. Lucambio Pérez and L. F. Prudente, "Nonlinear conjugate gradient methods for vector optimization," *SIAM Journal on Optimization*, vol. 28, no. 3, pp. 2690–2720, 2018.

[72] ——, "A Wolfe line search algorithm for vector optimization," *ACM Transactions on Mathematical Software*, vol. 45, no. 4, dec 2019.

[73] S. K. Mandal, D. Pacciarelli, A. LØkketangen, and G. Hasle, "A memetic NSGA-II for the bi-objective mixed capacitated general routing problem," *Journal of Heuristics*, vol. 21, no. 3, pp. 359–390, Jun. 2015, number: 3.

[74] P. Mansueto, "A memetic procedure for global multi-objective optimization," 2022, https://doi.org/10.5281/zenodo.7299857.

[75] ——, "Improved front steepest descent for multi-objective optimization," 2023, https://doi.org/10.5281/zenodo.7762661.

[76] ——, "LM-Q-NWT: A limited memory quasi-newton approach for multi-objective optimization," 2023, https://doi.org/10.5281/zenodo.7762660.

[77] ——, "Pareto front approximation through a multi-objective augmented lagrangian method," 2023, https://doi.org/10.5281/zenodo.8337581.

[78] P. Mansueto and M. Lapucci, "MOIHT & SFSD – Algorithms for cardinality-constrained multi-objective optimization problems," 2024, https://doi.org/10.5281/zenodo.10473331.

[79] P. Mansueto and F. Schoen, "Memetic differential evolution methods for clustering problems," *Pattern Recognition*, vol. 114, p. 107849, 2021.

[80] E. Miglierina, E. Molho, and M. Recchioni, "Box-constrained multi-objective optimization: A gradient-like method without "a priori" scalarization," *European Journal of Operational Research*, vol. 188, no. 3, pp. 662–682, 2008.

[81] S. Mostaghim, J. Branke, and H. Schmeck, "Multi-objective particle swarm optimization on computer grids," in *Proceedings of the 9th annual conference on Genetic and evolutionary computation*. ACM, 2007, pp. 869–875.

[82] B. K. Natarajan, "Sparse approximate solutions to linear systems," *SIAM Journal on Computing*, vol. 24, no. 2, pp. 227–234, 1995.

[83] J. Nocedal, "Updating quasi-Newton matrices with limited storage," *Mathematics of Computation*, vol. 35, no. 151, pp. 773–782, 1980.

[84] J. Nocedal and S. J. Wright, "Quasi-Newton Methods," in *Numerical Optimization*. New York, NY: Springer, 2006, pp. 135–163.

[85] A. Osyczka and S. Kundu, "A new method to solve generalized multicriteria optimization problems using the simple genetic algorithm," *Structural Optimization*, vol. 10, no. 2, pp. 94–99, Oct. 1995.

[86] G. Palermo, C. Silvano, S. Valsecchi, and V. Zaccaria, "A system-level methodology for fast multi-objective design space exploration," in *Proceedings of the 13th ACM Great Lakes symposium on VLSI*. ACM, 2003, pp. 92–95.

[87] A. Pascoletti and P. Serafini, "Scalarizing vector optimization problems," *Journal of Optimization Theory and Applications*, vol. 42, no. 4, pp. 499–524, 1984.

[88] R. Pellegrini, E. Campana, M. Diez, A. Serani, F. Rinaldi, G. Fasano, U. Iemma, G. Liuzzi, S. Lucidi, and F. Stern, "Application of derivative-free

multi-objective algorithms to reliability-based robust design optimization of a high-speed catamaran in real ocean environment1," *Engineering Optimization IV-Rodrigues et al.(Eds.)*, p. 15, 2014.

[89] Z. Povalej, "Quasi-Newton's method for multiobjective optimization," *Journal of Computational and Applied Mathematics*, vol. 255, pp. 765–777, Jan. 2014.

[90] L. Prudente and D. Souza, "A Quasi-Newton Method with Wolfe line searches for multiobjective optimization," *Journal of Optimization Theory and Applications*, vol. 194, no. 3, pp. 1107–1140, 2022.

[91] S. Qu, M. Goh, and F. T. S. Chan, "Quasi-Newton methods for solving multiobjective optimization," *Operations Research Letters*, vol. 39, no. 5, pp. 397–399, Sep. 2011.

[92] O. Schütze, A. Lara, and C. C. Coello, "The directed search method for unconstrained multi-objective optimization problems," *Proceedings of the EVOLVE–A Bridge Between Probability, Set Oriented Numerics, and Evolutionary Computation*, pp. 1–4, 2011.

[93] P. K. Shukla, "On gradient based local search methods in unconstrained evolutionary multi-objective optimization," in *International Conference on Evolutionary Multi-Criterion Optimization*. Springer, 2007, pp. 96–110.

[94] K. Sindhya, K. Miettinen, and K. Deb, "A hybrid framework for evolutionary multi-objective optimization," *IEEE Transactions on Evolutionary Computation*, vol. 17, no. 4, pp. 495–511, 2013.

[95] R. E. Steuer and E.-U. Choo, "An interactive weighted Tchebycheff procedure for multiple objective programming," *Mathematical Programming*, vol. 26, no. 3, pp. 326–344, 1983.

[96] Y. Sun, D. W. K. Ng, J. Zhu, and R. Schober, "Multi-objective optimization for robust power efficient and secure full-duplex wireless communication systems," *IEEE Transactions on Wireless Communications*, vol. 15, no. 8, pp. 5511–5526, 2016.

[97] H. Tanabe, E. H. Fukuda, and N. Yamashita, "Proximal gradient methods for multiobjective optimization and their applications," *Computational Optimization and Applications*, vol. 72, no. 2, pp. 339–361, Mar 2019.

[98] M. Tavana, "A subjective assessment of alternative mission architectures for the human exploration of Mars at NASA using multicriteria decision making," *Computers & Operations Research*, vol. 31, no. 7, pp. 1147–1164, 2004.

[99] A. M. Tillmann, D. Bienstock, A. Lodi, and A. Schwartz, "Cardinality minimization, constraints, and regularization: A survey," *arXiv preprint arXiv:2106.09606*, 2021.

[100] S. Tiwari, G. Fadel, P. Koch, and K. Deb, "Performance assessment of the hybrid archive-based micro genetic algorithm (amga) on the cec09 test problems," in *2009 IEEE Congress on Evolutionary Computation*, 2009, pp. 1935–1942.

[101] M. Villalobos-Cid, M. Dorn, R. Ligabue-Braun, and M. Inostroza-Ponta, "A memetic algorithm based on an NSGA-II scheme for phylogenetic tree inference," *IEEE Transactions on Evolutionary Computation*, vol. 23, no. 5, pp. 776–787, 2018.

[102] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 106, no. 1, pp. 25–57, 2006.

[103] X. Wang, C. Hirsch, S. Kang, and C. Lacor, "Multi-objective optimization of turbomachinery using improved NSGA-II and approximation model," *Computer Methods in Applied Mechanics and Engineering*, vol. 200, no. 9-12, pp. 883–895, 2011.

[104] J. Weston, A. Elisseeff, B. Schölkopf, and M. Tipping, "Use of the zero norm with linear models and kernel methods," *The Journal of Machine Learning Research*, vol. 3, pp. 1439–1461, 2003.

[105] D. White, "Epsilon-dominating solutions in mean-variance portfolio analysis," *European Journal of Operational Research*, vol. 105, no. 3, pp. 457–466, 1998.

[106] L. Zadeh, "Optimality and non-scalar-valued performance criteria," *IEEE transactions on Automatic Control*, vol. 8, no. 1, pp. 59–60, 1963.

[107] Q. Zhang, A. Zhou, S. Zhao, P. N. Suganthan, W. Liu, S. Tiwari *et al.*, "Multiobjective optimization test instances for the cec 2009 special session and competition," *University of Essex, Colchester, UK and Nanyang technological University, Singapore, special session on performance assessment of multi-objective optimization algorithms, technical report*, vol. 264, pp. 1–30, 2008.

[108] C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal, "Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization," *ACM Trans. Math. Softw.*, vol. 23, no. 4, pp. 550–560, dec 1997.

[109] E. Zitzler, K. Deb, and L. Thiele, "Comparison of multiobjective evolutionary algorithms: Empirical results," *Evol. Comput.*, vol. 8, no. 2, pp. 173–195, Jun. 2000.

[110] E. Zitzler and L. Thiele, "Multiobjective optimization using evolutionary algorithms — a comparative case study," in *Parallel Problem Solving from Nature — PPSN V*, A. E. Eiben, T. Bäck, M. Schoenauer, and H.-P. Schwefel, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998, pp. 292–301.