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

Convergent descent methods for smooth multiobjective optimization

Candidate
Guido Cocchi

Supervisors
Prof. Marco Sciandrone

Prof. Fabio Schoen

PhD Coordinator
Prof. Luigi Chisci

ciclo XXXI, 2015-2018

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

To my family

Acknowledgments

I would like to thank Professors Marco Sciandrone and Fabio Schoen for their human and scientific support which was very useful for my research activity. I would like also to thank them for letting me work in a nice place, the Global Optimization Laboratory (GOL), where a wonderful atmosphere walked with me through this three years. Moreover, I want to thank all the guys who have already taken the PhD title and the guys who will take it next years, for bearing me during this time: Luca, Niccolò, Federica, Alessandro, Francesco, Tommaso, Leonardo, Giulio, Luca, Leonardo, Alessio, Matteo.

A special thank is addressed to Giampaolo Liuzzi, who helped me, with great competence, for all the topics of the thesis.

Abstract

The thesis concerns the development of algorithms for smooth MultiObjective Optimization (MOO), where two or more objectives have to be simultaneously minimized. MOO plays a crucial role in several real-life applications like, for instance, portfolio selection problems, vehicle routing problems, traffic equilibrium problems. The main purpose of the work is to develop (in a rigorous way from a mathematical programming point of view) effective and efficient algorithms for smooth MOO problems based on a sound convergence theory. The first part of the thesis regards the design of algorithms for unconstrained MOO problems. Since the classical steepest descent algorithm generates a single Pareto-stationary point, a framework for the approximation of the Pareto front, using the steepest descent direction, is proposed. Convergence properties of the proposed algorithm are stated. The results of computational experiments performed on unconstrained MOO test problems have been reported.

In the second part of the thesis, sparse MOO problems are considered, i.e. problems where one of the objectives is the so-called zero-norm. The proposed approach is that of approximating the zero-norm by a smooth concave function. Equivalence properties between the original and the approximated problem have been stated and an algorithm based on the steepest descent framework has been designed, implemented and tested on sparse portfolio selection problems.

Finally, in the third part of the work, derivative-free MOO problems with box constraints have been considered. A method, that is a nontrivial extension of the well-known Implicit Filtering algorithm to the MOO case, is proposed. Global convergence results are stated under smooth assumptions on the objective functions. Numerical results on a set of test problems show the effectiveness of the MOO Implicit Filtering algorithm both when used to find a single Pareto solution of the problem and when used to reconstruct the whole Pareto front.

Contents

Chapter 1

Introduction

Many real-world applications do not fit well in the classic optimization framework in which only a single-objective function has to be minimized. In fact, we have to deal with two or more conflicting objectives to be simultaneously minimized. Such problems are known, in literature, as MultiObjective Optimization (MOO) problems [41].

The MOO framework has been widely used in literature for portfolio selection [4, 10, 16], vehicle routing [34, 44, 47], supply chain management problems [2, 49, 51] and design problems [26, 35, 43, 50].

A MOO problem can be expressed, in its general form, as follows:

$$\min_{x \in \mathcal{F} \subseteq \mathbb{R}^n} (f_1(x), \ldots, f_m(x)) \tag{1.1}$$

where $f_i : \mathcal{F} \to \mathbb{R}$, $\forall\ i \in \{1, \ldots, m\}$.

Given a problem of the form (1.1), since multiple mathematically equivalent global minima called *Pareto optima* can be retrieved, the interaction between the algorithm/solver and the decision maker (who chooses the best solution according to some preferences) plays a fundamental role.

For this reason, a list of multiobjective strategies depending on the relationship between the solver and the decision maker, is provided according to [41]:

- *Without preferences:* no preferences are set by the decision maker, hence each Pareto optimum is acceptable.
- *A priori:* the decision maker sets its preferences in advance, in order to let the solver retrieve the best possible solution.

- *A posteriori:* the whole Pareto optimal set has to be generated, since the decision maker will choose the best solution after a subsequent evaluation. The richer the set of final solutions presented to the decision maker, the more freedom the decision maker has when selecting its preferred solution.

- *Interactive:* Preferences are progressively provided during the algorithm execution, in order to address it to the best solution.

Regardless of the relationship between the solver and the decision maker, one of the most common approaches for MOO problems is the scalarization method in which one or more single-objective problems have to be separately solved. The most common scalarization method consists in a minimization of a weighted sum of the objectives [19, 33, 42].

MOO methods which do not scalarize the objectives are considered in this work.

Multiobjective steepest descent [20, 22], Newton [21] and Quasi-Newton [45] methods are *without preferences* extensions of their well-known single-objective versions, while other multiobjective algorithms are extensions of evolutionary and genetic algorithms for global optimization [14, 15, 28, 53].

Under the assumption of continuous differentiability of the objective functions, the multiobjective steepest descent algorithm computes, for each iteration k, the multiobjective steepest descent direction by solving the following quadratic programming problem:

$$\min_{v \in \mathbb{R}^n} \max_{i \in \{1,\dots,m\}} \nabla f_i(x_k)^\top v + \frac{1}{2}||v||^2. \tag{1.2}$$

Then, if the solution $v(x_k)$ of problem (1.2) is non-zero, a sufficient decrease for all the objectives can be obtained through suitable line search techniques along $v(x_k)$.

The first part of the work aims to propose an *a posteriori* extension of the steepest descent algorithm in which a sequence of sets $\{L_k\}$, composed of non dominated solutions, is generated.

In order to improve this list, for each iteration k, every point $x_c \in L_k$ is considered. Steepest descent directions $v(x_c)$ are computed by solving problems like 1.2. Then, two Armijo-type line search techniques able to exploit the directions $v(x_c)$ for generating new non-dominated solutions are proposed. Moreover, under some assumptions, global convergence properties of the algorithm in terms of sequence of sets are stated.

Numerical results on unconstrained multiobjective problems are reported in order to show the effectiveness of the proposed framework.

The second part of this work aims to obtain sparse solutions of problem (1.1), i.e. solutions with the highest number of components equal to zero.

For instance, in [31] simultaneous minimization of both the reconstruction error of a deep neural network and the sparsity of the vector of parameters that describe the network itself is considered. In the context of signal processing and reconstruction, [37] considers the multiobjective problem defined by the minimization of the measurement error along with a sparsity-inducing term. In [10], sparsity of an investor's portfolio is considered as a further objective in the classical mean-variance Markowitz approach to portfolio optimization. We thus consider multiobjective problems where one of the objective functions counts the number of non-zero components of the solution vector x, i.e. the following multiobjective sparse optimization problem:

$$\min_{x \in \mathcal{F} \subset \mathbb{R}^n} (f_1(x), \ldots, f_{m-1}(x), ||x||_0) \tag{1.3}$$

where the function $||\cdot||_0$ is the ℓ_0-norm which is equal to the number of non-zero component of x, and $\mathcal{F}$ is a compact convex set. Since sparse multiobjective optimization problems are combinatorial problems, we approximate the ℓ_0-norm with some smooth concave functions as in [46]. Equivalence properties in terms of Pareto optima between the problem (1.3) and its concave approximation are stated. Moreover, an *a posteriori* algorithm for problem (1.3), based on the steepest descent framework is proposed. Numerical results are obtained comparing the proposed framework with state-of-the-art multiobjective methods on real portfolio selection problems.

In the third part of this work, we consider problems like (1.1) with box constraints i.e.:

$$\min_{\ell \leq x \leq u} (f_1(x), \ldots, f_m(x)) \tag{1.4}$$

Moreover, all the objectives are *black-box* type, i.e. first (and higher) order derivatives cannot be directly computed nor approximated in any way. We remark that this situation is common in many applications, see e.g. [39].

In [12, 39], deterministic algorithms, which export derivative-free techniques for single-objective problems to the multiobjective context, have been proposed. Also the Implicit Filtering algorithm, originally proposed in [29], was widely applied to single-objective derivative-free optimal design prob-

lems [8, 11, 13, 18, 24, 25, 30]. Basically, the Implicit Filtering algorithm is a finite-difference gradient-based method in that it makes use of gradient approximations obtained by finite differences. On the one hand, even though the Implicit Filtering algorithm cannot be directly used to solve problem (1.4), we approach the problem with the multiobjective steepest descent method, but by approximating objective gradients as in the Implicit Filtering strategy. Hence, we define a new *without preferences* method for the solution of problem (1.4). The method exports the derivative-free skills of the Implicit Filtering approach [30, 36] within the multiobjective steepest descent framework. For this method, we also prove convergence to a Pareto-stationary point of problem (1.4). On the other hand, an *a posteriori* version of the algorithm is proposed. We also show how our multiobjective implicit filtering algorithm can be used within Direct-Multisearch algorithm [12] to improve its ability to generate the Pareto front of the problem (1.4).

1.1 Notation

With reference to problem (1.1), we denote by $F : \mathcal{F} \subseteq \mathbb{R}^n \to \mathbb{R}^m$ the vector-valued function defined by

$$F(x) \stackrel{\triangle}{=} (f_1(x), \ldots, f_m(x))^\top,$$

If F is a continuously differentiable map, we denote by $J : \mathcal{F} \subseteq \mathbb{R}^n \to \mathbb{R}^{m \times n}$ its Jacobian matrix function,

$$J(x) = (\nabla f_1(x), \ldots, \nabla f_m(x))^\top.$$

Given a point $x \in \mathcal{F}$, we define the set of feasible directions in x, namely $\mathcal{D}(x)$:

$$\mathcal{D}(x) := \{d \in \mathbb{R}^n : \exists \bar{t} > 0, \forall t \in (0, \bar{t}]\ x + td \in \mathcal{F}\} \tag{1.5}$$

If $\mathcal{F}$ is a convex set, then we can express the set $\mathcal{D}(x)$ as follows:

$$\mathcal{D}(x) := \{(y - x) : y \in \mathcal{F}\} \tag{1.6}$$

Given any two vectors $u, v \in \mathbb{R}^p$,

$$\begin{aligned} u < v &\quad \Leftrightarrow \quad u_i < v_i, \text{ for all } i = 1, \ldots, p \\ u \leqq v &\quad \Leftrightarrow \quad u_i \leq v_i, \text{ for all } i = 1, \ldots, p \\ u \leq v &\quad \Leftrightarrow \quad u \leqq v \text{ and } u \neq v. \end{aligned}$$

Finally, let us denote by $e_i \in \mathbb{R}^n$, $i = 1, \ldots, n$, the vectors that form the canonical basis in $\mathbb{R}^n$, and by $\mathbf{1}$ the vector, of appropriate dimension, of all ones, e.g., in $\mathbb{R}^n$, $\mathbf{1} = \sum_{i=1}^{n} e_i$.

1.2 Preliminary results

As in the single-objective optimization context, in the multiobjective case we should give a way to compare two points x, y with respect to their vector-valued functions $F(x), F(y)$.

Definition 1 (Pareto Dominance)**.** *Given $x, y \in \mathcal{F}$, we say that x strictly Pareto-dominates y when*

$$F(x) \leq F(y) \tag{1.7}$$

Then, with reference to the problem (1.1), an ideal solution would be a point x^* which dominates any other point i.e.:

$$F(x^*) \leqq F(x), \ \forall x \in \mathcal{F} \tag{1.8}$$

Unfortunately, that solution does not exist in general. So a new optimality definition has to be provided.

Definition 2 (Weak Pareto optimality)**.** *Given $x^* \in \mathcal{F}$, we say that x^* is a Weak-Pareto-optimum for problem* (1.1) *when*

$$F(x) \not< F(x^*), \ \forall x \in \mathcal{F} \tag{1.9}$$

Definition 3 (Pareto optimality)**.** *Given $x^* \in \mathcal{F}$, we say that x^* is a Pareto-optimum for problem* (1.1) *when*

$$F(x) \not\leq F(x^*), \ \forall x \in \mathcal{F} \tag{1.10}$$

By means of these two definitions, we are able to identify a set of non-dominated points (the so-called Pareto front or frontier) which is constituted by the *optimal* solutions of the multiobjective problem (1.1). Just as in the single-objective case, to define solution algorithms and analyze their convergence properties, we report from [27] the notion of Pareto-stationarity.

Definition 4 (Pareto-stationarity)**.** *Let the objective functions be continuously differentiable. Given $x^* \in \mathcal{F}$, we say that x^* is Pareto-stationary for problem* (1.1) *when, $\forall d$ in $\mathcal{D}(x^*)$, there exists an index $j \in \{1, \ldots, m\}$ such that:*

$$\nabla f_j(x^*)^\top d \geq 0, \tag{1.11}$$

It can be easily shown that, if x^* is Pareto-optimum, then x^* is Pareto-stationary (the inverse implication can only be shown when F is a strictly convex continuously differentiable map, as in the single-objective case).

1.2.1 Steepest descent for unconstrained multiobjective optimization

Let us consider the following unconstrained multiobjective optimization problem:

$$\min_{x \in \mathbb{R}^n} (f_1(x), \ldots, f_m(x)) \tag{1.12}$$

where all the objective functions are continuously differentiable.

Thanks to Definition 4, if $x^* \in \mathbb{R}^n$ is not a Pareto-stationary point, then there exists a descent direction v for all the objective functions f_i, $i \in \{1, \ldots, m\}$, at x^*.

In the unconstrained case, i.e. $\mathcal{D}(x) = \mathbb{R}^n$, for any given $x \in \mathbb{R}^n$, we define the function $g_x : \mathbb{R}^n \to \mathbb{R}$ by

$$g_x(v) = \max_{i \in \{1,\ldots,m\}} \nabla f_i(x)^\top v. \tag{1.13}$$

Note that g_x is continuous, piecewise linear, and convex.

Let us consider the following optimization problem:

$$\min_{v \in \mathbb{R}^n} g_x(v) + \frac{1}{2}||v||^2. \tag{1.14}$$

Note that the additional quadratic term $\frac{1}{2}||v||^2$ makes the problem well-defined. Hence, since the objective function is proper, closed and strongly convex, as reported in [27], problem (1.14) has always a (unique) optimal solution $v(x)$, which we call the steepest descent direction.

Definition 5 (Unconstrained steepest descent direction)**.** *Given any point $x \in \mathbb{R}^n$, the steepest descent direction for F at x is*

$$v(x) \in \arg\min_{v \in \mathbb{R}^n} g_x(v) + \frac{1}{2}||v||^2. \tag{1.15}$$

Let us consider the following function $\theta : \mathbb{R}^n \to \mathbb{R}$, which is the optimal value of (1.14):

$$\theta(x) = \min_{v \in \mathbb{R}^n} g_x(v) + \frac{1}{2}||v||^2. \tag{1.16}$$

Furthermore, from [22], we report the following proposition.

Proposition 1. *Given problem* (1.12)*, let* $v : \mathbb{R}^n \to \mathbb{R}^n$ *and* $\theta : \mathbb{R}^n \to \mathbb{R}$ *be defined as in* (1.15) *and* (1.16) *respectively. Then the following statements hold:*

- $\theta(x) \leq 0, \quad \forall x \in \mathbb{R}^n$
- $x^* \in \mathbb{R}^n$ *is Pareto-stationary for problem* (1.12) *iff* $\theta(x^*) = 0$
- *The mappings* $x \to v(x)$ *and* $x \to \theta(x)$ *are continuous mappings.*

1.2.2 Steepest descent for multiobjective optimization on convex sets

Let us consider the following multiobjective optimization problem:

$$\min_{x \in \mathcal{F}} \left(f_1(x), \ldots, f_m(x)\right) \tag{1.17}$$

where all the objectives are continuously differentiable and $\mathcal{F}$ is a compact convex set. Since $\mathcal{F}$ is convex, then for each $x \in \mathcal{F}$ we can express the set $\mathcal{D}(x)$ as in (1.6). In this case the function g_x, defined in (1.13), can be expressed as:

$$g_x(y) = \max_{i \in \{1,\ldots,m\}} \nabla f_i(x)^\top (y - x).$$

Due to the compactness of $\mathcal{F}$, there is no need to add any quadratic term as in the unconstrained case.

As in the unconstrained case, we can define the functions y and θ as follows:

$$y(x) = \arg\min_{y \in \mathcal{F}} g_x(y) \tag{1.18}$$

and

$$\theta(x) = \min_{y \in \mathcal{F}} g_x(y) \tag{1.19}$$

Since 1.18 is a min max problem, it may be conveniently transformed as follows:

$$\begin{aligned} &\min_{\tau \in \mathbb{R}, y \in \mathcal{F}} \tau \\ &\nabla f_i(x)^\top (y - x) - \tau \leq 0, \quad \forall i \in \{1, \ldots, m\} \end{aligned} \tag{1.20}$$

As in the unconstrained case, from [22], we report the following proposition.

Proposition 2. *Given problem* (1.17)*, let* $y : \mathcal{F} \to \mathcal{F}$ *and* $\theta : \mathcal{F} \to \mathbb{R}$ *be defined as in* (1.18) *and* (1.19) *respectively. Then the following statements hold:*

- $\theta(x) \leq 0, \quad \forall x \in \mathcal{F}$
- $x^* \in \mathcal{F}$ *is Pareto-stationary for Problem* (1.17) *iff* $\theta(x^*) = 0$
- *The mappings* $x \to y(x)$ *and* $x \to \theta(x)$ *are continuous mappings.*

We introduce the steepest descent direction for the vector valued mapping F at x.

Definition 6 (Steepest descent direction on convex sets)**.** *Given any point* $x \in \mathcal{F}$*, the steepest descent direction for* F *at* x *is*

$$v(x) = y(x) - x \tag{1.21}$$

where $y(x)$ *is given by* (1.18)*.*

1.2.3 The steepest descent algorithm for multiobjective optimization

On the basis of Definitions (5) and (6), from reference [22], we recall the following steepest descent algorithm for the solution of Problems (1.12) and (1.17) (we recall from the introduction that $J(x)$ denotes the Jacobian of the vector of objective functions).

At each iteration k, the steepest descent direction $v(x_k)$ is computed. If a sufficient condition of Pareto-stationarity is met, then the algorithm stops, otherwise an Armijo-type line search along $v(x_k)$ is done. As in the single-objective case, the line search stops when a sufficient decrease condition with respect to all the objectives is reached.

Some convergence properties of Algorithm 1 in the unconstrained case will be recalled in the next subsection, but we refer the reader to [22] for further details.

Algorithm 1: `steepest_descent_algorithm`

input: $x_0 \in \mathcal{F} \subseteq \mathbb{R}^n$, $\gamma \in (0, 1)$
output: a Pareto-stationary point x^*
for $k = 0, 1, \ldots,$ **do**
    Compute $\theta(x_k)$ and $v(x_k)$
    if $\theta(x_k) = 0$ **then**
        $x^* \leftarrow x_k$
        return // x^* `is Pareto-stationary`
    end
    Compute $\alpha_k = 2^{-\beta_k}$ with β_k the smallest non-negative integer s.t.

$$F(x_k + \alpha_k v(x_k)) \leq F(x_k) + \gamma \alpha_k J(x_k) v(x_k).$$

    $x_{k+1} \leftarrow x_k + \alpha_k v(x_k)$
end

1.2.4 Convergence properties of multiobjective steepest descent algorithm

Let us refer to the unconstrained multiobjective optimization problem (1.12) where $f_i : \mathbb{R}^n \to \mathbb{R}$ are continuously differentiable.

The steepest descent algorithm, defined in the previous subsection (see Algorithm 1), updates the current point when a condition of sufficient decrease with respect to all the objective functions is reached. So, it generates a sequence of points with Pareto-stationarity convergence properties.

Global convergence properties of the steepest descent algorithm (see [22, 27] for further details) are reported below.

Proposition 3 (Convergence of the multiobjective steepest descent algorithm: the general case)**.** *Let $\{x_k\}$ be the sequence generated by Algorithm 1. Then every limit point, if any, is a Pareto-stationary point.*

Since Algorithm 1 produces a globally convergent sequence of points $\{x_k\}$, in Chapter 2 we will define a globally convergent *a posteriori* extension able to generate a sequence of sets of non-dominated points.

1.2.5 Metrics for Pareto front evaluation

Although it is easy to evaluate the performance of *without preferences* solvers, for example counting how many times a solvers retrieve a better point in the sense of Definition 1, when two or more *a posteriori* solvers have to be compared, a set of metrics able to evaluate the capability of a given solver to obtain a good Pareto front in terms of both optimality and diversification is needed.

Three metrics were proposed in [12]: Purity, Spread Γ and Spread Δ. From now on, let $\mathcal{P}$ be the set of multiobjective optimization problems and $\mathcal{S}$ be the set of the considered solvers. Since the true Pareto front is not always provided for each $p \in \mathcal{P}$, its best possible approximation F_p^{true} is computed by firstly merging all the retrieved Pareto fronts i.e.:

$$\tilde{F}_p^{true} := \bigcup_{s \in \mathcal{S}} F_{p,s} \tag{1.22}$$

where $F_{p,s}$ is the approximation of the Pareto front computed by solver s with respect to the objective space and after by removing from $\tilde{F}_p^{true}$ every

dominated solution that is

$$F_p^{true} = \{\, x \in \tilde{F}_p^{true} : \nexists y \in \tilde{F}_p^{true} \ s.t.\ F(y) \leq F(x) \,\}. \tag{1.23}$$

On the basis of F_p^{true}, we can define in a rigorous way the proposed metrics.

Purity

Purity metric, first introduced in [9], is able to establish the accuracy of a given solver in terms of Pareto optimality.

Given a problem $p \in \mathcal{P}$, remembering that $F_{p,s}$ is the approximation of the Pareto front computed by solver s and F_p^{true} is its best possible approximation, Purity metric is defined as follows:

$$P_{p,s} := \frac{|F_{p,s} \cap F_p^{true}|}{|F_{p,s}|}. \tag{1.24}$$

As we can see from (1.24), this metric represents the "precision" of a solver to retrieve Pareto-optimal points. The higher Purity coefficient, the higher probability of retrieving Pareto-optimal points is.

Spread metrics

The most relevant drawback concerning the Purity metric is that the size of the retrieved Pareto front does not have an impact. In fact, if two solvers compute two Pareto fronts composed of 100 and 1 non-dominated solutions respectively, they have the same Purity value. Hence, we need some metrics able to take into account the diversification of a Pareto front.

Spread metrics were introduced in [12] in order to deal with this problem.

Given a problem $p \in \mathcal{P}$, remembering that F_p^{true} is the best possible approximation of its Pareto front, for each objective $j \in \{1, \ldots, m\}$, we first compute the "extreme points" as follows:

$$f_j^0 := \min_{f_j \in F_p^{true}} f_j \tag{1.25}$$

$$f_j^{N+1} := \max_{f_j \in F_p^{true}} f_j. \tag{1.26}$$

Such points will be the same for the application of the metrics on any of the obtained fronts.

Then, given a solver s and its Pareto front approximation $F_{p,s}$, let us suppose to sort ascending the list of N retrieved points $f^{i,s}$, with respect to each objective j. Then Spread Γ metric can be defined as follows:

$$\Gamma_{p,s} := \max_{j\in\{1,\dots,m\}} \max_{i\in\{0,\dots,N\}} \underbrace{f_j^{i+1,s} - f_j^{i,s}}_{\delta_j^{i,s}} \tag{1.27}$$

where $f_j^{0,s} = f_j^0$ and $f_j^{N+1,s} = f_j^{N+1}, \forall s \in \mathcal{S}$.

In other words, given a Pareto front approximation, Spread Γ metric measures its maximum "hole" size in the objective space. Please note that $\Gamma_{p,s} \geq 0$, since the retrieved points were previously sorted ascending with respect to the objective j. Note that with $m = 2$, this metric is simply the maximum distance in the infinity norm between consecutive points in the approximated Pareto front.

Another spread metric was proposed in [12] in order to assess how well the points are distributed in the Pareto front approximation.

On the basis of the definition of $\delta_j^{i,s}$ in the (1.27), Spread Δ metric can be defined as follows:

$$\Delta_{p,s} := \max_{j\in\{1,\dots,m\}} \frac{\delta_j^{0,s} + \delta_j^{N,s} + \sum_{i=1}^{N-1} |\delta_j^{i,s} - \bar{\delta_j^s}|}{\delta_j^{0,s} + \delta_j^{N,s} + (N-1)\bar{\delta_j^s}} \tag{1.28}$$

where $\bar{\delta_j^s}$ is the mean "hole" size in the Pareto front retrieved by solver s, with respect to the objective j, i.e.:

$$\bar{\delta_j^s} := \frac{1}{N-1} \sum_{i=1}^{N-1} \delta_j^{i,s} \tag{1.29}$$

As we can see from (1.28), Spread Δ is quite similar to the standard deviation of the "hole" sizes, so it evaluates the uniformity of a Pareto front approximation.

Chapter 2

A globally convergent a posteriori algorithm based on the steepest descent framework

The multiobjective steepest descent method [20, 22] is a natural extension of its single-objective formulation, with properties of global convergence to Pareto-stationary points. In this part of the work, an *a posteriori* generalization of the steepest descent method is proposed. Since a sequence of sets of non-dominated points is generated, at every iteration, each point of the current set is exploited in order to compute the steepest descent direction by solving problems like (1.2). Two different Armijo-type line search techniques, proven to generate new non-dominated solutions, are proposed. For each type of line search, convergence properties of the algorithm in terms of the list of points are stated. Numerical results, obtained on a test set composed of unconstrained multiobjective optimization problems, show the effectiveness of the proposed framework.

2.1 The algorithm

In this section, we are interested in the definition of an algorithm that produces a set of non-dominated points, i.e. an approximation of the Pareto front for a multiobjective problem like (1.12) where all the objective functions are continuously differentiable. To this aim, the proposed algorithm

(see Algorithm 2) produces a sequence of sets of points (rather than a sequence of points as usual in the single-objective case) namely $\{L_k\}$. More in particular, for each iteration k, a finite set of non-dominated points L_k can be defined as:

$$L_k = \{x_j \in \mathbb{R}^n,\ j = 1, \ldots, r_k\} \tag{2.1}$$

where $r_k = |L_k|$. At each iteration k, each point $x_c \in L_k$ is considered for improving the current list. Then, the steepest descent direction in x_c is computed by solving the following problem:

$$\min_{v \in \mathbb{R}^n} \max_{i \in \{1,\ldots,m\}} \nabla f_i(x_c)^\top v + \frac{1}{2}||v||^2. \tag{2.2}$$

Let $v(x_c)$ be the solution of problem (2.2), then the value of $\theta(x_c)$ can be computed as follows:

$$\theta(x_c) = \max_{i \in \{1,\ldots,m\}} \nabla f_i(x_c)^\top v(x_c) + \frac{1}{2}||v(x_c)||^2. \tag{2.3}$$

Since the traditional Armijo-type line search does not fit well our aim (see Algorithm 1 line 9), we propose a modification of the standard line search algorithm (see Algorithm 3) in order to find a point which is sufficiently non-dominated by the current list $\tilde{L}_k$.

We now prove that Algorithm 3 is able to add a new non-dominated solution thanks to the following proposition.

Proposition 4. *Let $x_c \in \tilde{L}_k$ be such that $\theta(x_c) < 0$, i.e. $v(x_c)$ exists such that*

$$\nabla f_s(x_c)^\top v(x_c) + \frac{1}{2}||v(x_c)||^2 < 0$$

$\forall s \in \{1, \ldots, m\}$. Then $\exists \alpha > 0$, sufficiently small, such that

$$F(x_j) + \mathbf{1}\gamma\alpha\theta(x_c) \not\leq F(x_c + \alpha v(x_c)), \quad \forall x_j \in \tilde{L}_k,$$

i.e. the while loop of Algorithm 3 terminates in a finite number of iterations.

Proof. First, let us recall that $\tilde{L}_k$ is a finite set of non-dominated points. Then, since $x_c \in \tilde{L}_k$, there cannot exist any $x_j \in \tilde{L}_k$ such that

$$F(x_j) \leq F(x_c), \tag{2.4}$$

that is to say that x_j dominates x_c. Now, we proceed by contradiction, and assume that, for all $h = 1, 2, \ldots$, a point $x_{j_h} \in \tilde{L}_k$ exists such that

$$F(x_{j_h}) + \mathbf{1}\gamma\delta^h\Delta\theta(x_c) < F(x_c + \delta^h\Delta v(x_c)).$$

Algorithm 2: `front_steepest_descent_algorithm`

input: an initial set of non-dominated points L_0.
output: a final list of non-dominated points L^*.
$k \leftarrow 0$
while *stopping criterion not satisfied* **do**
  $\tilde{L}_k \leftarrow L_k$
  for $c = 1, \ldots, r_k$ **do**
    if $x_c \in \tilde{L}_k$ **then**
      Compute $v(x_c)$ and $\theta(x_c)$
      if $\theta(x_c) < 0$ **then**
        $\alpha \leftarrow$ `armijo_type_line_search`$(x_c, v(x_c), \theta(x_c), \tilde{L}_k)$
        $x_{new} \leftarrow x_c + \alpha v(x_c)$
        $\tilde{L}_k \leftarrow \{x_j \in \tilde{L}_k \mid F(x_{new}) \not\preceq F(x_j)\} \cup \{x_{new}\}$
      end
    end
  end
  $L_{k+1} \leftarrow \tilde{L}_k$
  $k \leftarrow k + 1$
end
$L^* \leftarrow L_k$
return L^*

Algorithm 3: `armijo_type_line_search`

input: $x_c \in \tilde{L}_k$, $v(x_c)$, $\theta(x_c)$, $\tilde{L}_k$, $\Delta > 0$, $\delta \in (0,1)$, $\gamma \in (0,1)$
output: an optimal stepsize α
$\alpha \leftarrow \Delta$
while $\exists\ x_j \in \tilde{L}_k$ *s.t.* $F(x_j) + \mathbf{1}\gamma\alpha\theta(x_c) < F(x_c + \alpha v(x_c))$ **do**
  $\alpha \leftarrow \delta\alpha$
end
return α

Since $|\tilde{L}_k|$ is finite, it is possible to consider a subsequence $H \subseteq \{1, 2, \dots\}$ such that $j_h = \bar{\jmath}$, for all $h \in H$. Hence, for all $h \in H$, we would have

$$F(x_{\bar{\jmath}}) + \mathbf{1}\gamma\delta^h\Delta\theta(x_c) < F(x_c + \delta^h\Delta v(x_c)). \tag{2.5}$$

On the other hand, since $v(x_c)$ is a descent direction for all f_s with $s \in \{1, \dots, m\}$, we know that, for $h \in H$ and sufficiently large, it must result

$$f_s(x_c + \delta^h\Delta v(x_c)) < f_s(x_c) + \gamma\delta^h\Delta\theta(x_c), \quad \forall s \in \{1, \dots, m\}. \tag{2.6}$$

Then, by (2.5) and (2.6), we could write

$$\begin{aligned} f_s(x_{\bar{\jmath}}) + \gamma\delta^h\Delta\theta(x_c) &< f_s(x_c + \delta^h\Delta v(x_c)) \\ &< f_s(x_c) + \gamma\delta^h\Delta\theta(x_c), \quad \forall s \in \{1, \dots, m\}, \end{aligned}$$

that is

$$f_s(x_{\bar{\jmath}}) < f_s(x_c), \quad \forall s \in I. \tag{2.7}$$

Now, by taking the limit for $h \to \infty, h \in H$ in (2.5), we would obtain

$$F(x_{\bar{\jmath}}) \leqq F(x_c),$$

which, recalling (2.4), would imply $F(x_{\bar{\jmath}}) = F(x_c)$, thus contradicting relation (2.7) and concluding the proof.

□

2.2 Convergence analysis

In order to establish the convergence properties of the algorithm, a definition of *linked sequences* is reported from [39].

Definition 7. *Let $\{L_k\}$ be the sequence of sets of non-dominated points produced by Algorithm 2. We define a* linked sequence *as a sequence $\{x_{j_k}\}$ such that, for any $k = 1, 2, \ldots$, $x_{j_k} \in L_k$ is generated at iteration $k-1$ of Algorithm 2-3 by the point $x_{j_{k-1}} \in L_{k-1}$.*

Hence, $x_{j_k} = x_{j_{k-1}} + \alpha_{j_k} v(x_{j_{k-1}})$ and it results

$$F(x_{\ell_j}) + \mathbf{1}\gamma\alpha_{j_k}\theta(x_{j_{k-1}}) \not< F(x_{j_k})$$

for all $x_{\ell_j} \in L_{k-1}$.

Then we introduce the following assumption needed to prove the convergence results of Algorithm 2.

Assumption 1. *There exists a point $x_0 \in L_0$ such that the set*

$$\mathcal{L}_0 = \bigcup_{i=1}^{m} \left\{ x \in \mathbb{R}^n : f_i(x) \leq f_i(x_0) \right\}$$

is compact.

Proposition 5. *Let us assume that Assumption 1 holds.*
Let $\{L_k\}$ be the sequence of sets of non-dominated points produced by the Algorithm 2. Let $\{x_{j_k}\}$ be a linked sequence. Then it admits limit points and every limit point is Pareto-stationary for problem (1.12).

Proof. First of all, let $x_{j_0} \in L_0$ be the point for which Assumption 1 holds. We first show that the every linked sequence $\{x_{j_k}\}$ admits a limit point $\bar{x}$. The steps of Algorithm 2 guarantee that

$$F(x_0) \not< F(x_{j_k}),\ \forall k$$

and that $\forall k$ there exists an index i_{j_k} such that

$$x_{j_k} \in \left\{ x \in \mathbb{R}^n : f_{i_{j_k}}(x) \leq f_{i_{j_k}}(x_0) \right\}.$$

Therefore we can conclude that

$$x_{j_k} \in \mathcal{L}_0,\ \forall k.$$

Then Assumption 1 ensures that the linked sequence $\{x_{j_k}\}$ is bounded.

Now, in order to state the main result we first prove that, for every linked sequence $\{x_{j_k}\}$, we have:

$$\lim_{k\to\infty} \alpha_{j_k}\theta(x_{j_{k-1}}) = 0. \tag{2.8}$$

In fact, let us assume, by contradiction, that there exists a set K such that:

$$-\alpha_{j_k}\theta(x_{j_{k-1}}) \geq \delta > 0 \qquad \forall k \in K. \tag{2.9}$$

Since $\{x_{j_k}\}$ is a linked sequence, for all $x_{\ell_j} \in L_{k-1}$ and $k \in K$, it results

$$F(x_{\ell_j}) + \mathbf{1}\gamma\alpha_{j_k}\theta(x_{j_{k-1}}) \not< F(x_{j_k}). \tag{2.10}$$

By using (2.9) we obtain that, for all $x_{\ell_j} \in L_{k-1}$ and $k \in K$,

$$F(x_{\ell_j}) - \mathbf{1}\gamma\delta \not< F(x_{j_k}). \tag{2.11}$$

This implies that the infinite points $F(x_{j_k}) \in Z$, with $k \in K$, have a distance not smaller than $\gamma\delta$ from each other and that, hence, the set

$$Z = \{z \in \mathbb{R}^m : \ z = F(x), \ \forall x \in \mathcal{L}_0\}$$

is not compact. This last point contradicts Assumption 1 and the continuity assumptions of the functions f_i, $i = 1, \ldots, m$. Therefore we obtain that (2.8) holds.

Now we recall that $\bar{x}$ is Pareto-stationary for problem (1.12) if and only is $\theta(\bar{x}) = 0$ (see Proposition 1). Assume, by contradiction, that a limit point $\bar{x}$ of a linked sequence $\{x_{j_k}\}$ is not Pareto-stationary. This is equivalent to say that there exist a scalar $\varepsilon > 0$ and a set K such that

$$\theta(x_{j_k}) \leq -\varepsilon < 0, \qquad \forall k \in K \tag{2.12}$$

which, by using (2.8), yields that

$$\lim_{k\to\infty, k\in K} \alpha_{j_k} = 0. \tag{2.13}$$

Therefore, for sufficiently large values of k, we have that $\alpha_{j_k} < \Delta$. Then the steps of Algorithm 3 and the definition of L_{k-1} imply that there exists $x_{h_{j_k}} \in L_{k-1}$ and $k \in K$, such that

$$F(x_{h_{\ell_j}}) + \mathbf{1}\gamma\frac{\alpha_{j_k}}{\delta}\theta(x_{j_{k-1}}) < F(x_{j_{k-1}} + \frac{\alpha_{j_k}}{\delta}v(x_{j_{k-1}})) \tag{2.14}$$

$$F(x_{h_{\ell_j}}) \not< F(x_{j_{k-1}}) \tag{2.15}$$

Now (2.14) and (2.15) yield that an index $s_k \in \{1, \dots, m\}$ exists such that

$$f_{s_k}(x_{j_{k-1}}) + \gamma \frac{\alpha_{j_k}}{\delta} \theta(x_{j_{k-1}}) < f_{s_k}(x_{j_{k-1}} + \frac{\alpha_{j_k}}{\delta} v(x_{j_{k-1}}))$$

namely

$$f_{s_k}(x_{j_{k-1}} + \frac{\alpha_{j_k}}{\delta} v(x_{j_{k-1}})) - f_{s_k}(x_{j_{k-1}}) > \gamma \frac{\alpha_{j_k}}{\delta} \theta(x_{j_{k-1}})$$

Now, since $s_k \in \{1, \dots, m\}$ we can consider a subset $\tilde{K} \subseteq \bar{K}$ such that, for all $k \in \tilde{K}$, $s_k = \bar{s}$, so that

$$f_{\bar{s}}(x_{j_{k-1}} + \frac{\alpha_{j_k}}{\delta} v(x_{j_{k-1}})) - f_{\bar{s}}(x_{j_{k-1}}) > \gamma \frac{\alpha_{j_k}}{\delta} \theta(x_{j_{k-1}})$$

By the Mean-value Theorem, we have that

$$f_{\bar{s}}(x_{j_{k-1}} + \frac{\alpha_{j_k}}{\delta} v(x_{j_{k-1}})) - f_{\bar{s}}(x_{j_{k-1}}) = \frac{\alpha_{j_k}}{\delta} \nabla f_{\bar{s}}(\xi_{j_{k-1}})^\top v(x_{j_{k-1}})$$

with

$$\xi_{j_{k-1}} = x_{j_{k-1}} + t_{j_{k-1}} \frac{\alpha_{j_k}}{\delta} v(x_{j_{k-1}}), \quad t_{j_{k-1}} \in (0, 1).$$

Then, we can write

$$\nabla f_{\bar{s}}(\xi_{j_{k-1}})^\top v(x_{j_{k-1}}) \geq \gamma \theta(x_{j_{k-1}}).$$

The definition of function θ gives

$$\theta(x_{j_{k-1}}) + (\nabla f_{\bar{s}}(\xi_{j_{k-1}}) - \nabla f_{\bar{s}}(x_{j_{k-1}}))^\top v(x_{j_{k-1}}) \geq \gamma \theta(x_{j_{k-1}}).$$

Then we have

$$(1 - \gamma)\theta(x_{j_{k-1}}) + (\nabla f_{\bar{s}}(\xi_{j_{k-1}}) - \nabla f_{\bar{s}}(x_{j_{k-1}}))^\top v(x_{j_{k-1}}) \geq 0$$

By taking the limit for $k \to \infty$ and $k \in \tilde{K}$, by recall that $\alpha_{j_k} \to 0$ and by considering the boundedness of $v(x_{j_{k-1}})$ and the continuity of θ, we obtain the contradiction

$$(1 - \gamma)\theta(\bar{x}) \geq 0$$

which concludes the proof.

□

2.3 The Armijo-type extrapolation technique

In order to improve the capability of the Algorithm 2 of spanning the space, we introduce a new Armijo-type line search technique (see Algorithm 4). This new line search differs from Algorithm 3 by the following points:

- the initial stepsize can vary at every iteration;
- if the initial point satisfies the acceptability criterion, the algorithm performs an extrapolation along the considered direction;
- the algorithm can produce more than one point.

The following proposition states that Algorithm 4 is well defined.

Proposition 6. *Let $x_c \in \tilde{L}_k$ be such that $\theta(x_c) < 0$, i.e. $v(x_c)$ exists such that*

$$\nabla f_s(x_c)^\top v(x_c) + \frac{1}{2}||v(x_c))||^2 < 0, \quad \forall s \in \{1, \dots, m\}. \tag{2.16}$$

Then Algorithm 4 is well defined, namely it cannot infinitely cycle, and returns a not empty list of steps α^*.

Proof. In case of the first if-instruction of the algorithm being satisfied, a similar reasoning to the one used in in the proof of Proposition 4 proves that the first while loop of Algorithm 4 cannot infinitely cycle, so the list α^* is updated once.

Therefore, by contradiction, we assume that the second while-loop of the algorithm infinitely cycles, i.e., a monotonically increasing sequence of positive numbers $\{\alpha^h\}$ exists such that:

$$F(x_j) + \mathbf{1}\gamma\frac{\alpha^h}{\delta}\theta(x_c) \not< F(x_c + \frac{\alpha^h}{\delta}v(x_c)), \qquad \forall x_j \in \tilde{L}_k \cup L_{tmp},$$

and, in particular if $x_j = x_c$,

$$F(x_c) + \mathbf{1}\gamma\frac{\alpha^h}{\delta}\theta(x_c) \not< F(x_c + \frac{\alpha^h}{\delta}v(x_c)). \tag{2.17}$$

The previous (2.17) implies that an index $s_k \in \{1, \dots, m\}$ exists such that

$$f_{s_k}(x_c) + \gamma\frac{\alpha^h}{\delta}\theta(x_c) \geq f_{s_k}(x_c + \frac{\alpha^h}{\delta}v(x_c)).$$

Algorithm 4: `armijo_type_line_search with extrapolation`

input: $x_c \in \tilde{L}_k$, $v(x_c)$, $\theta(x_c)$, $\tilde{L}_k$, $\Delta_k > 0$, $\delta \in (0,1)$, $\gamma \in (0,1)$
output: a sequence of steps α^*
$\alpha \leftarrow \Delta_k$
if $\exists\ x_j \in \tilde{L}_k\ s.t\ F(x_j) + \mathbf{1}\gamma\alpha\theta(x_c) < F(x_c + \alpha v(x_c))$ **then**
  while $\exists\ x_j \in \tilde{L}_k\ s.t\ F(x_j) + \mathbf{1}\gamma\alpha\theta(x_c) < F(x_c + \alpha v(x_c))$ **do**
    $\alpha \leftarrow \delta\alpha$
  end
  $\alpha^* \leftarrow \{\alpha\}$
else
  $L_{tmp} \leftarrow \emptyset$
  while $F(x_j) + \mathbf{1}\gamma\frac{\alpha}{\delta}\theta(x_c) \not< F(x_c + \frac{\alpha}{\delta}v(x_c)),\ \forall\ x_j \in \tilde{L}_k \cup L_{tmp}$ **do**
    if $F(x_c + \alpha v(x_c)) + \mathbf{1}\gamma\frac{1-\delta}{\delta}\alpha\theta(x_c) \not> F(x_c + \frac{\alpha}{\delta}v(x_c))$ **then**
      $\alpha^* \leftarrow \alpha^* \cup \{\alpha\}$
      $L_{tmp} \leftarrow L_{tmp} \cup \{x_c + \alpha v(x_c)\}$
    end
    $\alpha \leftarrow \frac{\alpha}{\delta}$
  end
  if $\alpha = \Delta_k$ **then**
    $\alpha^* \leftarrow \alpha^* \cup \{\alpha\}$
  end
end
return α^*

Recalling, again, that $s_k \in \{1, \ldots, m\}$, we can consider a subset K, such that, for all $k \in K$, it results $s_k = \bar{s}$, so that

$$f_{\bar{s}}(x_c) + \gamma\frac{\alpha^h}{\delta}\theta(x_c) \geq f_{\bar{s}}(x_c + \frac{\alpha^h}{\delta}v^(x_c)).$$

This relation and the fact that $\alpha^h \to \infty$ contradict Assumption 1.

Now we prove that, during the second while-loop, the lists L_{tmp} and α^* are updated at least once.
Since the second while loop terminates in a finite number of steps an index $\bar{h}$ and a point $\tilde{x}_j \in \tilde{L}_k \cup L_{tmp}$ exist such that

$$F(\tilde{x}_j) + \mathbf{1}\gamma\frac{\alpha_k}{\delta^{\bar{h}}}\theta(x_c) < F(x_c + \frac{\alpha_k}{\delta^{\bar{h}}}v(x_c)). \tag{2.18}$$

Namely, the test of the while loop is not satisfied. Instead, at the previous step, the test is verified:

$$F(x_j) + \mathbf{1}\gamma\frac{\alpha_k}{\delta^{\bar{h}-1}}\theta(x_c) \not< F(x_c + \frac{\alpha_k}{\delta^{\bar{h}-1}}v(x_c)), \qquad \forall x_j \in \tilde{L}_k \cup L_{tmp}.$$

This means that for $\tilde{x}_j \in \tilde{L}_k$ an index $s_j \in \{1, \ldots, m\}$ exists such that:

$$f_{s_j}(\tilde{x}_j) + \gamma\frac{\alpha_k}{\delta^{\bar{h}-1}}\theta(x_c) \geq f_{s_j}(x_c + \frac{\alpha_k}{\delta^{\bar{h}-1}}v(x_c)). \tag{2.19}$$

On the other hand (2.18) yields:

$$f_{s_j}(\tilde{x}_j) + \gamma\frac{\alpha_k}{\delta^{\bar{h}}}\theta(x_c) < f_{s_j}(x_c + \frac{\alpha_k}{\delta^{\bar{h}}}v(x_c)). \tag{2.20}$$

By combining (2.19) and (2.20) it is possible to obtain:

$$f_{s_j}(x_c + \frac{\alpha_k}{\delta^{\bar{h}-1}}v(x_c)) + \gamma(1-\delta)\frac{\alpha_k}{\delta^{\bar{h}}}\theta(x_c) < f_{s_j}(x_c + \frac{\alpha_k}{\delta^{\bar{h}}}v(x_c)).$$

This inequality shows that the if-condition in the second while loop is satisfied (by setting $\alpha = \frac{\alpha_k}{\delta^{\bar{h}-1}}$) and hence the lists L_{tmp} and α^* are updated.

If the while-condition is never satisfied, i.e. a point $\tilde{x}_j \in \tilde{L}_k$ exists such that

$$F(\tilde{x}_j) + \mathbf{1}\gamma\frac{\Delta_k}{\delta}\theta(x_c) < F(x_c + \frac{\Delta_k}{\delta}v(x_c)), \tag{2.21}$$

then, the list α^* is updated due to the if-condition below the while loop.

□

Since Algorithm 4 generates a sequence of steps along direction $v(x_c)$, in order to correctly update the list $\tilde{L}_k$, the line 10 of Algorithm 2 has to be changed as follows:

---

for $\alpha_i \in \alpha$ **do**
  $\tilde{L}_k \leftarrow \{x_j \in \tilde{L}_k \mid F(x_c + \alpha_i v(x_c)) \not\leq F(x_j)\} \cup \{x_c + \alpha_i v(x_c)\}$
end

Finally we prove the global convergence properties of Algorithm 2-4.

Proposition 7. *Let us assume that Assumption 1 holds.*
Let $\{L_k\}$ be the sequence of sets of non-dominated points produced by the Algorithm 2-4. Let $\{x_{j_k}\}$ be a linked sequence. Then it admits limit points and every limit point is Pareto-stationary for problem (1.12).

Proof. First of all, let $x_{j_0} \in L_0$ be the point for which Assumption 1 holds. The steps of Algorithm 4 ensure that the points of every linked sequence $\{x_{j_k}\}$ of Algorithm 2-4 satisfy the property that, for all $x_{\ell_j} \in L_{k-1}$

$$F(x_{\ell_j}) + \mathbf{1}\gamma\alpha_{j_k}\theta(x_{j_{k-1}}) \not< F(x_{j_k}). \tag{2.22}$$

By using Assumption 1, the definition of L_k, property (2.22) and by repeating the same reasoning of the first part of the proof of Proposition 5, it is possible to prove that the sequence $\{L_k\}$ is bounded and

$$\lim_{k\to\infty} \alpha_{j_k}\theta(x_{j_{k-1}}) = 0. \tag{2.23}$$

Now we assume, by contradiction, that a limit point $\bar{x}$ of a linked sequence $\{x_{j_k}\}$ is not Pareto-stationary and that, hence, there exist a scalar $\varepsilon > 0$ and a set K such that such that

$$\theta(x_{j_{k-1}}) \leq -\varepsilon, \qquad \forall k \in K, \tag{2.24}$$

which, by using (2.23), yields that

$$\lim_{k\to\infty, k\in K} \alpha_{j_k} = 0. \tag{2.25}$$

Now the if-test in the second while-loop implies that for all $k \in K$

$$F(x_{j_k} + \frac{1-\delta}{\delta}\alpha_{j_k} v(x_{j_{k-1}})) \not< F(x_{j_k}) + \mathbf{1}\gamma\frac{1-\delta}{\delta}\alpha_{j_k}\theta(x_{j_{k-1}})$$

which yields that an index $s_k \in \{1, \dots, m\}$ exists such that

$$f_{s_k}(x_{j_k} + \frac{1-\delta}{\delta}\alpha_{j_k} v(x_{j_{k-1}})) \geq f_{s_k}(x_{j_k}) + \gamma \frac{1-\delta}{\delta}\alpha_{j_k}\theta(x_{j_{k-1}})\mathbf{1}$$

namely

$$f_{s_k}(x_{j_k} + \frac{1-\delta}{\delta}\alpha_{j_k} v(x_{j_{k-1}})) - f_{s_k}(x_{j_k}) \geq \gamma \frac{1-\delta}{\delta}\alpha_{j_k}\theta(x_{j_{k-1}}).$$

Recalling that $s_k \in \{1, \dots, m\}$, we can consider a subset $\tilde{K} \subseteq \bar{K}$ such that, for all $k \in \tilde{K}$, $s_k = \bar{s}$, so that

$$f_{\bar{s}}(x_{j_k} + \frac{1-\delta}{\delta}\alpha_{j_k} v(x_{j_{k-1}})) - f_{\bar{s}}(x_{j_k}) \geq \gamma \frac{1-\delta}{\delta}\alpha_{j_k}\theta(x_{j_{k-1}})$$

By the Mean-value Theorem, we have that

$$f_{\bar{s}}(x_{j_k} + \frac{1-\delta}{\delta}\alpha_{j_k} v(x_{j_{k-1}})) - f_{\bar{s}}(x_{j_k}) = \frac{1-\delta}{\delta}\alpha_{j_k} \nabla f_{\bar{s}}(\xi_{j_k})^\top v(x_{j_{k-1}})$$

with

$$\xi_{j_k} = x_{j_k} + t_{j_k}\frac{1-\delta}{\delta}\alpha_{j_k} v(x_{j_{k-1}}), \quad t_{j_k} \in (0,1).$$

Then, we can write

$$\nabla f_{\bar{s}}(\xi_{j_k})^\top v(x_{j_{k-1}}) \geq \gamma\theta(x_{j_{k-1}}).$$

The definition of function θ gives

$$\theta(x_{j_{k-1}}) + (\nabla f_{\bar{s}}(\xi_{j_k}) - \nabla f_{\bar{s}}(x_{j_{k-1}}))^\top v^{\bar{I}}(x_{j_{k-1}}) \geq \gamma\theta(x_{j_{k-1}}).$$

Then we have

$$(1-\gamma)\theta(x_{j_{k-1}}) + (\nabla f_{\bar{s}}(\xi_{j_k}) - \nabla f_{\bar{s}}(x_{j_{k-1}}))^\top v(x_{j_{k-1}}) \geq 0$$

By taking the limit for $k \to \infty$ and $k \in \tilde{K}$, by recalling that $\alpha_{j_k} \to 0$ and by considering the boundedness of $v(x_{j_{k-1}})$ and the continuity of θ, we obtain the contradiction

$$(1-\gamma)\theta(\bar{x}) \geq 0$$

which concludes the proof.

□

2.4 Preliminary numerical results

In this section, we report some preliminary numerical results in order to assess the effectiveness of the proposed framework.

The aim is to show how the proposed framework performs (`FRONT-SD`) with respect to a multistart version of the *without preferences* steepest descent algorithm (`MULTISTART-SD`), which iteratively calls the Algorithm 1 starting from different randomly sampled points.

Test problems: we considered the set of 10 different unconstrained multiobjective problems used for the Cec2009 competition [52] whose dimension n is equal to 30 and with a number m of objectives belonging to the set $\{2, 3\}$. In order to increase the dataset size, we vary the number of variables from 5 to 50 with step 5, obtaining a set of 100 problems.

Implementation details: we implemented Algorithm 2 in Python 3.6 using Tensorflow 1.5 [1] for computing derivatives and Gurobi [32] for solving quadratic programming problems like (2.2). Armijo extrapolation technique (see Algorithm 4) is used as a line search step with the following additional parameters:

$$\begin{aligned} &\Delta_k = 1, \qquad \delta = 0.5 \\ &\gamma = 10^{-5}. \end{aligned}$$

The algorithm stops when one of the two following stopping conditions is reached:

- maximum number of function evaluations (a Jacobian matrix evaluation costs n).
- all points $x_c \in L_k$ have been explored.

For our experiments, the maximum number of function evaluations is set to 20,000.

Since a box-constrained version for each problem is given in [52], then the centroid of the hyperbox $[\ell, u]$ is chosen as starting point i.e. $L_0 = \{x_0\}$ with x_0 such that:

$$(x_0)_i = \frac{\ell_i + u_i}{2} \quad \forall i \in \{1, \dots, n\}. \tag{2.26}$$

Since some test problems contain objective functions that are not defined everywhere, infinite values are assigned to the singularities, while points of non-differentiability are not considered for space exploration.

Purity, Spread Γ and Spread Δ metrics defined in section 1.2.5, have been used with the performance profiles benchmarking technique [17] for comparing the performance.

We recall that the Purity metric measures the quality of the generated front, i.e. how good the non-dominated points computed by a solver are with respect to those computed by any other solver. On the other hand, the Spread metrics are essential to measure the uniformity of the generated front in the objectives space.

Note that, for each problem p, the "reference" Pareto front F_p^{true} is calculated by first computing

$$\tilde{F}_p^{true} = F_{p,\texttt{FRONT-SD}} \cup F_{p,\texttt{MULTISTART-SD}},$$

where $F_{p,s}$ denotes the set of non-dominated solutions found by solver s, and then removing from this set any dominated solution as in (1.23).

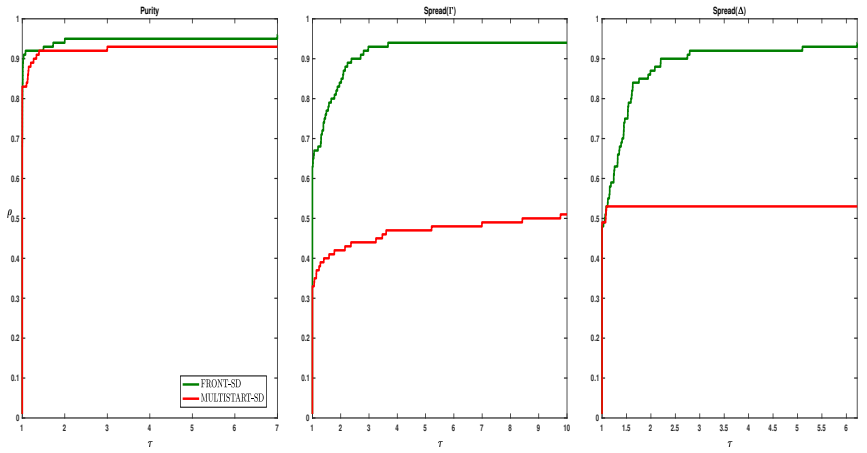

Figure 2.1: Performance profiles of Purity and Spread metrics for `FRONT-SD` and `MULTISTART-SD`.

Figure 2.1 shows the effectiveness of our framework, namely `FRONT-SD`, with respect to the `MULTISTART-SD`. In particular, `FRONT-SD` highly outperforms `MULTISTART-SD` with respect to Spread metrics. This is an expected behaviour, since `MULTISTART-SD` does not build iteratively a list of non-dominated solutions, but it executes several independent runs starting from different points resulting in a non uniform Pareto front.

Chapter 3

Multiobjective methods with a cardinality constraint through concave approximation

On top of the difficulty of dealing with multiple objectives, in many applications one is also particularly interested in obtaining "sparse" solutions of (1.1), i.e. solutions with the highest number of components equal to zero.

We thus consider Problem (1.17) where one of the objective functions counts the number of non-zero components of the solution vector x, i.e. the following multiobjective sparse optimization problem

$$\min_{x \in \mathcal{F} \subset \mathbb{R}^n} \ (f_1(x), f_2(x), \ldots, f_{m-1}(x), ||x||_0) \tag{SMOP}$$

where $m \geq 2$, $\mathcal{F} \subset \mathbb{R}^n$ is a compact convex set, and $\|x\|_0$ is the ℓ_0-norm of x, namely,

$$\|x\|_0 = \Big|\big\{i \in \{1, \ldots, n\} : \ |x_i| > 0\big\}\Big|.$$

Problem (SMOP) is effectively a combinatorial problem and it is thus considerably more difficult [3] than (1.17). In order to make the problem more tractable, and following the approach of [46], in this paper we propose to address the ℓ_0-norm objective by means of smooth concave approximating functions. In this way, we are able to convert (SMOP) into a smooth problem. Further, we prove that such an approximating problem enjoys a nice equivalence property with respect to the original combinatorial one. Then,

we define an algorithm based on the steepest descent framework for smooth multiobjective optimization.

3.1 Concave approximation approaches

As stated in the introduction section, (SMOP) is a difficult combinatorial problem. This section is devoted to the description and analysis of convenient ways to approximate the (discontinuous) zero-norm function in Problem (SMOP). In particular, we consider smooth concave approximating functions to convert (SMOP) into a smooth problem. This approach is motivated by the fact that

$$\|x\|_0 = \sum_{i=1}^{n} s(|x_i|),$$

where $s : \mathbb{R} \to \mathbb{R}^+$ is the step function such that $s(t) = 1$ for $t > 0$ and $s(t) = 0$ for $t \leq 0$. The idea is then to replace the discontinuous step function with a continuously differentiable concave function.

To this purpose, and drawing inspiration from [46], we first rewrite Problem (SMOP) as follows:

$$\begin{aligned} &\min_{x \in \mathcal{F} \subset \mathbb{R}^n, y \in \mathbb{R}^n} \ (f_1(x), f_2(x), \ldots, f_{m-1}(x), ||y||_0) \\ &-y \leq x \leq y, \end{aligned} \tag{3.1}$$

where inequality $-y \leq x \leq y$ is intended componentwise, i.e. $-y_i \leq x_i \leq y_i$, $i = 1, \ldots, n$.

In the following, we study the equivalence between problem (3.1) and a problem of the form

$$\begin{aligned} &\min_{x \in \mathcal{F}, y \in \mathbb{R}^n} \ F(x, y) = \left(f_1(x), \ldots, f_{m-1}(x), \sum_{i=1}^{n} f^u(y_i) \right) \\ &-y \leq x \leq y \end{aligned} \tag{3.2}$$

where $f^u : \mathbb{R}^+ \to \mathbb{R}$ is a smooth function depending on a parameter $u \in U \subseteq \mathbb{R}$.

To this aim, we introduce the following assumption on the parameterized function f^u.

Assumption 2. *There exists $\bar{u} \in U$ such that, for any infinite sequence $\{u_k\} \to \bar{u}$ we have that:*

(i) for each $y_i \geq 0$, $\lim_{k\to\infty} f^{u_k}(y_i)$ *is well defined;*

(ii) for each $y_i > 0$, *it follows* $f^{u_k}(0) < f^{u_k}(y_i)$ *and*

$$\lim_{k\to\infty} f^{u_k}(0) < \lim_{k\to\infty} f^{u_k}(y_i) < \infty;$$

(iii) for any $\bar{y}_i > 0$, *and for any sequence* $\{y_i^k\} \to \bar{y}_i$, *we have*

$$\lim_{k\to\infty} f^{u_k}(y_i^k) = \lim_{k\to\infty} f^{u_k}(\bar{y}_i);$$

(iv) for each $y_i \geq 0$, *one of the following conditions holds: either*

$$\lim_{k\to\infty} f^{u_k}(y_i) = \begin{cases} 1 & \text{if } y_i > 0, \\ 0 & \text{if } y_i = 0, \end{cases} \tag{3.3}$$

or

$$\lim_{k\to\infty} f^{u_k}(0) = -\infty. \tag{3.4}$$

It can be shown that, when $U = \mathbb{R}^+$, Assumption 2 is satisfied, for instance:

- by $f^u(y_i) = 1 - e^{-uy_i}$, with $\bar{u} = +\infty$, which satisfies condition (3.3);
- by $f^u(y_i) = \log(u + y_i)$, with $\bar{u} = 0$, which satisfies condition (3.4).

In particular, we note that, whenever condition (3.3) holds, it results

$$\lim_{k\to\infty} \sum_{i=1}^{n} f^{u_k}(y_i) = \|y\|_0. \tag{3.5}$$

Now, concerning the connections between Problem (3.2) and Problem (SMOP), the following proposition holds.

Proposition 8. *Let* $\{u_k\}$ *be a sequence such that* $\lim_{k\to\infty} u_k = \bar{u}$ *and let* $\{(x^k, y^k)\}$ *be a sequence such that,* (x^k, y^k) *is weakly Pareto optimal for Problem (3.2) with* $u = u^k$, *and* $y^k = |x^k|$. *Then,* $\{(x^k, y^k)\}$ *has limit points and every limit point* $(\bar{x}, \bar{y})$ *is such that* $\bar{x}$ *is weakly Pareto optimal for Problem* (SMOP).

Proof. By the assumptions, for all k it results $x^k \in \mathcal{F}$ and $y^k = |x^k|$. Hence, as $\mathcal{F}$ is compact, the sequence $\{(x^k, y^k)\}$ admits limit points. We proceed by contradiction and assume that there exists an infinite subset $K \subseteq \{1, 2, \ldots\}$ such that

$$\lim_{k \in K, k \to \infty} (x^k, y^k) = (\bar{x}, \bar{y}),$$

and $\bar{x}$ is not weakly Pareto optimal for Problem (SMOP). Then, there must exist a point $v \in X$ such that

$$F(v) < F(\bar{x}). \tag{3.6}$$

Recalling the continuity of functions $f_1, \ldots, f_{m-1}$, we get that, for $k \in K$ sufficiently large, the following inequalities hold:

$$f_i(v) < f_i(x^k) \qquad i = 1, \ldots, m-1. \tag{3.7}$$

Moreover, from (3.6), recalling that $\bar{y} = |\bar{x}|$, it follows that

$$||v||_0 < ||\bar{x}||_0 = ||\bar{y}||_0. \tag{3.8}$$

Let y^v be such that $y^v = |v|$. Then, (v, y^v) is feasible for problem (3.2). Therefore, recalling that (x^k, y^k) is weakly Pareto-optimal for Problem (3.2) and taking (3.7) into account, for $k \in K$ sufficiently large we must have that

$$\sum_{i=1}^{n} f^{u_k}(y_i^k) \leq \sum_{i=1}^{n} f^{u_k}(y_i^v) = \sum_{i=1}^{n} f^{u_k}(|v_i|). \tag{3.9}$$

Consider any $i \in \{1, \ldots, n\}$ such that $\bar{y}_i > 0$. From assumption (iii), it follows that

$$\lim_{k \to \infty} f^{u_k}(\bar{y}_i) = \lim_{k \to \infty} f^{u_k}(y_i^k) = l_i. \tag{3.10}$$

Then, given any positive ϵ such that $n\epsilon < 1$, two positive integers $k_1(\epsilon)$ and $k_2(\epsilon)$ exist such that

$$f^{u_k}(\bar{y}_i) \leq l_i + \frac{\epsilon}{2}, \quad \text{for all } k \geq k_1(\epsilon),$$

$$f^{u_k}(y_i^k) \geq l_i - \frac{\epsilon}{2}, \quad \text{for all } k \geq k_2(\epsilon).$$

Thus, for k sufficiently large, we obtain

$$f^{u_k}(\bar{y}_i) \leq f^{u_k}(y_i^k) + \epsilon. \tag{3.11}$$

Now, let us consider any index $i \in \{1, \ldots, n\}$ such that $\bar{y}_i = 0$. Using assumption (ii), we have, for all k,

$$f^{u_k}(\bar{y}_i) \leq f^{u_k}(y_i^k). \tag{3.12}$$

From (3.11) and (3.12), we get that for k sufficiently large, we can write

$$\sum_{i=1}^{n} f^{u_k}(\bar{y}_i) \leq \sum_{i=1}^{n} f^{u_k}(y_i^k) + n\epsilon. \tag{3.13}$$

Condition (3.9) implies that

$$\sum_{i=1}^{n} f^{u_k}(\bar{y}_i) \leq \sum_{i=1}^{n} f^{u_k}(y_i^v) + n\epsilon = \sum_{i=1}^{n} f^{u_k}(|v_i|) + n\epsilon. \tag{3.14}$$

Let us now distinguish two cases.

Case I: Suppose that condition (3.3) holds. Using (3.5), we have

$$\lim_{k\to\infty} \sum_{i=1}^{n} f^{u_k}(\bar{y}_i) = \|\bar{y}\|_0 = \|\bar{x}\|_0,$$

$$\lim_{k\to\infty} \sum_{i=1}^{n} f^{u_k}(y_i^v) = \|y^v\|_0 = \|v\|_0.$$

Hence, taking limits for $k \to \infty$ in (3.14), we obtain

$$\|\bar{y}\|_0 \leq \|v\|_0 + n\epsilon.$$

From the above relation and (3.8), it follows

$$\|v\|_0 + 1 \leq \|\bar{y}\|_0 \leq \|v\|_0 + n\epsilon,$$

which contradicts the fact that $n\epsilon < 1$.

Case II: Suppose that condition (3.4) holds. First, we rewrite relation (3.14) as follows:

$$\sum_{\bar{y}_i>0} f^{u_k}(\bar{y}_i) + (n - \|\bar{y}\|_0) f^{u_k}(0) \leq \sum_{y_i^v>0} f^{u_k}(y_i^v) + (n - \|y^v\|_0) f^{u_k}(0) + n\epsilon,$$

from which we obtain

$$(\|y^v\|_0 - \|\bar{y}\|_0) f^{u_k}(0) \leq \sum_{y_i^v>0} f^{u_k}(y_i^v) - \sum_{\bar{y}_i>0} f^{u_k}(\bar{y}_i) + n\epsilon.$$

Taking limits for $k \to \infty$, using (3.8), the fact that $\|y^v\|_0 = \|v\|_0$ and condition (3.4), we get that the left member of the above relation tends to $+\infty$, while the right member tends to a finite value (see assumption (ii)), a contradiction. □

Remark 1. *In general, it turns out that weak Pareto optimality is the best we can aim for, even if we consider a sequence $\{(x^k, y^k)\}$ of Pareto points for the approximated problem, instead of the weak Pareto optimality used in Proposition (8). To have a better understanding of this point, we provide the following example.*

Example 1. *Consider the following multiobjective optimization problem:*

$$\begin{aligned} &\min_{x \in \mathbb{R}} (-x, \|x\|_0) \\ &-1 \leq x \leq 1. \end{aligned}$$

It is easy to see that the only Pareto optimal points are $x = 0$ and $x = 1$, whose objective vectors are $(0, 0)$ and $(-1, 1)$, respectively.

Consider now the sequence of smooth approximating problems

$$\begin{aligned} &\min_{x,y \in \mathbb{R}} (-x, f^{u_k}(y)) \\ &-1 \leq x \leq 1, \\ &-y \leq x \leq y, \end{aligned} \tag{P_{u_k}}$$

indexed by the sequence $\{u_k\}$. Consider, for example, the function $f^{u_k}(y) = 1 - e^{-u_k y}$. It is easy to see that, for any chosen value of $u_k > 0$, the point $(\frac{1}{2}, \frac{1}{2})$ is Pareto optimal for Problem (P_{u_k}). Then, consider the sequence $\{(x^k, y^k)\}$ such that $x^k = \frac{1}{2}$ and $y^k = \frac{1}{2}$ for every k. Taking limits for $k \to \infty$, $x^k \to \bar{x} = \frac{1}{2}$, which is not Pareto optimal for the original problem, since its objective vector is $F(x) = (-\frac{1}{2}, 1)$. However, $\bar{x}$ is a weak Pareto optimal point for the original problem.

The same reasoning still holds if we consider the function $f^{u_k}(y) = \log(u_k + y)$.

3.2 The algorithm

Inspired by the work in [23], in this section we propose an algorithm to approximate the Pareto front of Problem (SMOP). The idea, which is the same of Chapter 2, is to iteratively improve a list of non-dominated points.

First, let us denote

$$f_m(y) = \sum_{i=1}^{n} f^u(y_i),$$

so that Problem (3.2) can be rewritten as

$$\begin{aligned} &\min\ F(x,y) = \ (f_1(x), \ldots, f_{m-1}(x), f_m(y)) \\ &x \in \mathcal{F} \\ &-y \leq x \leq y. \end{aligned} \tag{3.15}$$

From now on, we consider $z = (x, y)^\top$ as the set of optimization variables.

Our algorithm is composed of three fundamental phases: an initialization phase, a search phase and a refining phase.

Initialization: the list has to be initialized with a set of non-dominated solutions. For simplicity, we consider the case in which the list is initialized with a singleton. Hence, given a feasible point z_0 and a stepsize $\alpha_0 > 0$, we initialize the list $Z_0 = \{(z_0, \alpha_0)\}$.

Search phase: we try to improve the set of non-dominated points, iterating over the list Z_k. At every iteration, we select a pair (z, α) and we generate the following set of points:

$$S(z, \alpha) = \{(z + \alpha d_i, \alpha_0) \mid \nabla f_i(z)^T d_i < 0, i \in 1 \ldots, m\} \tag{3.16}$$

where each direction d_i is a feasible descent direction in z for the corresponding objective function f_i. Then, the list Z_{k+1} is updated only considering non-dominated solutions of the set $Z_k \cup S(z, \alpha)$. If $Z_{k+1} = Z_k$, i.e. no new points are added to the list, then the iteration is considered unsuccessful and the stepsize related to the point z is decreased by a factor $\delta < 1$. The rationale is that of spreading a set of initial points by separately considering the single objective functions. The strategy may be considered as a sort of "guided" multistart.

Refining phase: in order to drive the obtained non-dominated points towards the Pareto front, the multiobjective steepest descent strategy is applied for each point of the list Z_k.

A few comments are in order:

- The search phase stops when one of the following criteria is satisfied: all the stepsizes related to the list Z_k are lower than a positive tolerance or a maximum number of function evaluations is reached[1].

[1] A computation of the vector $F(x)$ counts as a single function evaluation. A computation of the gradient vector $\nabla F(x)$ counts as n function evaluations.

- For the generation of the set $S(z, \alpha)$, we actually consider two different feasible descent directions per objective, computed with the Projected Gradient and Frank-Wolfe method respectively. Moreover, we also include the common steepest descent direction, which, as described in equation 1.20, can be computed by retrieving the solution (τ^*, x^*, y^*) of the problem

$$\begin{aligned} &\min_{\tau,x,y} \ \tau \\ &\nabla f_i(\bar{x})^\top (x - \bar{x}) - \tau \leq 0, \quad i = 1, \ldots, m-1, \\ &\nabla f_m(\bar{y})^\top (y - \bar{y}) - \tau \leq 0, \\ &x \in \mathcal{F}, \\ &-y \leq x \leq y, \end{aligned} \tag{3.17}$$

  and setting $d_z = (d_x, d_y) = (x^* - \bar{x}, y^* - \bar{y})$.

- The notion of Pareto-dominance is referred to the original objective vector $(f_1(x), f_2(x), \ldots, f_{m-1}(x), ||x||_0)^\top$.

Algorithm 5: MultiObjective Sparse Optimization (MOSO)

input: a nonempty, finite set of pairs $Z_0 = \{(z_0, \alpha_0)\}$, where every z_0 is a non-dominated point, $\alpha_0 > 0$, $\delta < 1$ and $\gamma < 1$.
ouput: a Pareto front approximation Z_f.
Set $k \leftarrow 0$.
`// Start Search phase`
while *stopping criterion not satisfied* **do**
  Select a pair $(z, \alpha) \in Z_k$.
  Compute the set of points $S(z, \alpha)$ as in (3.16).
  Set `iter_success` $\leftarrow false$.
  if $S(z, \alpha) \neq \emptyset$ **then**
    Set $L \leftarrow Z_k \cup S(z, \alpha)$.
    Set $L_{tmp} \leftarrow \{(w, \alpha_w) \in L \mid \nexists \bar{z} \in L \ s.t. \ F(\bar{z}) \leq F(w)\}$.
    if $L_{tmp} \neq Z_k$ **then**
      Set $Z_{k+1} \leftarrow L_{tmp}$.
      `iter_success` $\leftarrow true$.
    end
  else
    $L_{tmp} \leftarrow Z_k$.
  end
  if not `iter_success` **then**
    Set $Z_{k+1} \leftarrow L_{tmp} \setminus \{(z, \alpha)\} \cup \{(z, \delta \cdot \alpha)\}$.
  end
  Set $k \leftarrow k + 1$.
end
`// End Search phase`
Set $Z_f \leftarrow \emptyset$.
`// Start Refining phase`
for *each* $z \in Z_k$ **do**
  $z^* \leftarrow$ `steepest_descent_algorithm`(z, γ).
  $Z_f \leftarrow Z_f \cup \{z^*\}$.
end
`// End Refining phase`

3.3 Computational experiments on sparse portfolio problems

To assess the effectiveness of our approach, we consider the following portfolio selection problem:

$$\begin{aligned} &\min_{x\in\mathbb{R}^n} \left(-\frac{\mu^\top x}{x^\top Q x}, \|x\|_0 \right) \\ &\mathbf{1}^\top x = 1 \\ &0 \le x \le u \end{aligned} \tag{3.18}$$

where $\mu \in \mathbb{R}^n$ and $Q \in \mathbb{R}^{n\times n}$ are the vector of expected returns and covariance matrix, respectively, and u is a resource constraint. This problem is obtained from the standard Markowitz model

$$\begin{aligned} &\min_{x\in\mathbb{R}^n} x^\top Q x \\ &\mathbf{1}^\top x = 1, \\ &\mu^\top x \ge R, \\ &0 \le x \le u \end{aligned} \tag{3.19}$$

replacing the objective function $x^\top Qx$ with the so called "Sharpe ratio" (which takes into account both the variance $x^\top Qx$ and the expected return $\mu^\top x$), and by adding $\|x\|_0$ as a further objective. In our experiments, we set $u = \mathbf{1}$.

Test problems: the data used in the following experiments consists of daily data for securities from the FTSE 100 index, from 01/2003 to 12/2007. Such data is public and available from the website http://www.bolsapt.com. The three data sets are referred to as DTS1, DTS2, and DTS3, and are formed by 12, 24, and 48 securities, respectively. The assets we considered for the generation of the datasets are those used in [10].

Implementation details: an approximation of the zero-norm with logarithmic functions was used. In particular

$$||x||_0 \approx \sum_{i=1}^{n} \log(\varepsilon + x_i) \tag{3.20}$$

Parameters of algorithm `MOSO` have been set as follows:

$$\begin{aligned} &\varepsilon = 10^{-5}, \quad &&\delta = 0.5 \\ &\alpha_0 = 1, &&\alpha_{min} = 10^{-7} \\ &\gamma = 10^{-6} \end{aligned}$$

where α_{min} represents the tolerance on the stepsizes for the search phase.

We compare the performance of `MOSO` with the Direct-Multisearch (`DMS`) algorithm [12] with its setup defined in [10]. Moreover we set the maximum number of function evaluations to $2,000,00$ and we consider the variables with absolute value lower than 10^{-8} as zero.

As a final note, we stress the fact that the scope of this section is the comparison between the "non-smooth" derivative-free approach, on which DMS is based, with the steepest descent approach founded on a smooth reformulation that our algorithm exploits. Note, however, that the results we report in the following sections are with respect to the original vector of objectives of problem (SMOP), i.e., considering the actual $\|x\|_0$ objective. The smooth reformulation only serves as a means to obtain a set of points on which we compute the original objective vector F, constructing an approximation of the Pareto front of problem (SMOP).

3.3.1 Single point comparison

Here, we show a comparison on the "single point" scenario, i.e., when only one Pareto point has to be retrieved. Since the goal is to obtain a single Pareto point, the list Z_k is always composed of a single solution. Moreover, we modify the search phase in `MOSO` as follows: we update the current solution only if the algorithm is able to find a point that strictly dominates it in the set L. More formally, lines 9-14 of Algorithm 5 are replaced by the the following:

---

1 $L \leftarrow S(z, \alpha)$
2 **for** *each* $\hat{z} \in L$ **do**
3 | **if** $\hat{z}$ *strictly dominates* Z_k **then**
4 | | $Z_k \leftarrow \hat{z}$
5 | | `iter_success` $\leftarrow$ *true*
6 | **end**
7 **end**

	MOSO-Wins	DMS-Wins
DTS1	**36**	3
DTS2	**95**	1
DTS3	**70**	6
FF10	**92**	7
FF17	**94**	6
FF48	**90**	10

Table 3.1: Single point comparison.

In Table 3.1, we report the results obtained on the DTS1, DTS2, DTS3, FF10, FF17 and FF48 datasets. We run both MOSO and DMS starting from 100 different initial points (each time, we use the same initial point for both algorithms). The list L_{init} of initial points is the following:

$$L_{init} = \{e_1, \ldots, e_n, x_{n+1}, \ldots, x_{100}\},$$

where $e_i \in \mathbb{R}^n$ is the i-th unit vector[2] defined in section 1.2, and $x_i \in \mathbb{R}^n$ are random dense feasible points. For each dataset, we report:

- the number of times the point obtained by MOSO strictly dominates the one obtained by DMS (column MOSO-Wins);
- the number of times the point obtained by DMS strictly dominates the one obtained by MOSO (column DMS-Wins).

The results show the effectiveness of MOSO. In fact, MOSO is able to outperform DMS on every dataset, dominating the DMS solution considerably often.

3.3.2 Pareto front comparison

In order to compare the two solvers, we compute the Purity and Spread metrics defined in section 1.2.5 for each problem. We recall that the Purity metric measures the quality of the generated front, i.e. how good the non-dominated points computed by a solver are with respect to those computed by any other solver. On the other hand, the Spread metrics are essential to measure the uniformity of the generated front in the objectives space. Note

[2]For the DTS1, DTS2 and DTS3, n is equal to 12, 24, 48, respectively. For the FF datasets, n is equal to 10, 17 and 48, respectively.

Purity		
Prob/Alg	MOSO	DMS
DTS1	**1**	0.17
DTS2	**1**	**1**
DTS3	**0.94**	0.22
FF10	**1**	0.5
FF17	**1**	0.5
FF48	**1**	**1**

Spread Γ		
Prob/Alg	MOSO	DMS
DTS1	2.65	**2.11**
DTS2	**3.29**	12
DTS3	**3.29**	9
FF10	**1**	**1**
FF17	**1**	**1**
FF48	**1**	5

Spread Δ		
Prob/Alg	MOSO	DMS
DTS1	0.77	**0.62**
DTS2	**0.70**	∞
DTS3	0.85	**0.78**
FF10	0.89	**0.52**
FF17	0.68	**0.5**
FF48	**0.93**	∞

Table 3.2: Purity and Spread tables for DTS and FF problems starting with a singleton.

that, for each problem p, the "reference" Pareto front F_p^{true} is calculated by first computing

$$\tilde{F}_p^{true} = F_{p,\texttt{DMS}} \cup F_{p,\texttt{MOSO}},$$

where $F_{p,s}$ denotes the set of non-dominated solutions found by solver s, and then removing from this set any dominated solution as in (1.23).

The solvers were initialized with the singleton $\{x_0\}$ such that:

$$x_0 = \begin{bmatrix} 0 & \dots & 0 & 1 \end{bmatrix}^\top. \tag{3.21}$$

In table 3.2, Purity and Spread metrics are reported for the three considered problems.

The obtained results show the effectiveness of our solver with respect to the Purity metric. In fact, due to the search directions computed with the Jacobian matrix, MOSO is able to generate better points until the list is composed only by Pareto-stationary points. With respect to the Spread Γ

metric, it is easy to see that when `MOSO` outperforms `DMS` the performance ratio is very high (see tables 3.2(b), DTS2 and DTS3 datasets), while when it is outperformed, the performance ratio is low.

With respect to the Spread Δ metric, a degenerate case happens for both the DTS2 and FF48 datasets. In fact, `DMS` is not able to extend the Pareto front. Because of the computation of Spread delta requires the mean distance between consecutive points in the front, it can be set to ∞ because only one point is retrieved. In the other problems, `DMS` outperforms `MOSO`.

In order to obtain a more robust comparison, we tested the two solvers starting from a list of 5 random points for each feasible cardinality (e.g., with $n = 12$ variables the starting list contains $12 \cdot 5 = 60$ points). We executed the algorithms with 5 different random seeds obtaining 30 total instances.

We compute the performance profiles [17] of the two solvers considering all the 30 instances of the problems.

As we can see in figure 3.2, no degenerate cases happen with random initializations. The profiles confirmed the results obtained with the singleton initialization. In fact, our solver outperforms `DMS` in terms of Purity and Spread Γ as in the singleton case. For the Spread Δ metric, once avoided degenerate cases, `DMS` algorithm outperforms `MOSO`.

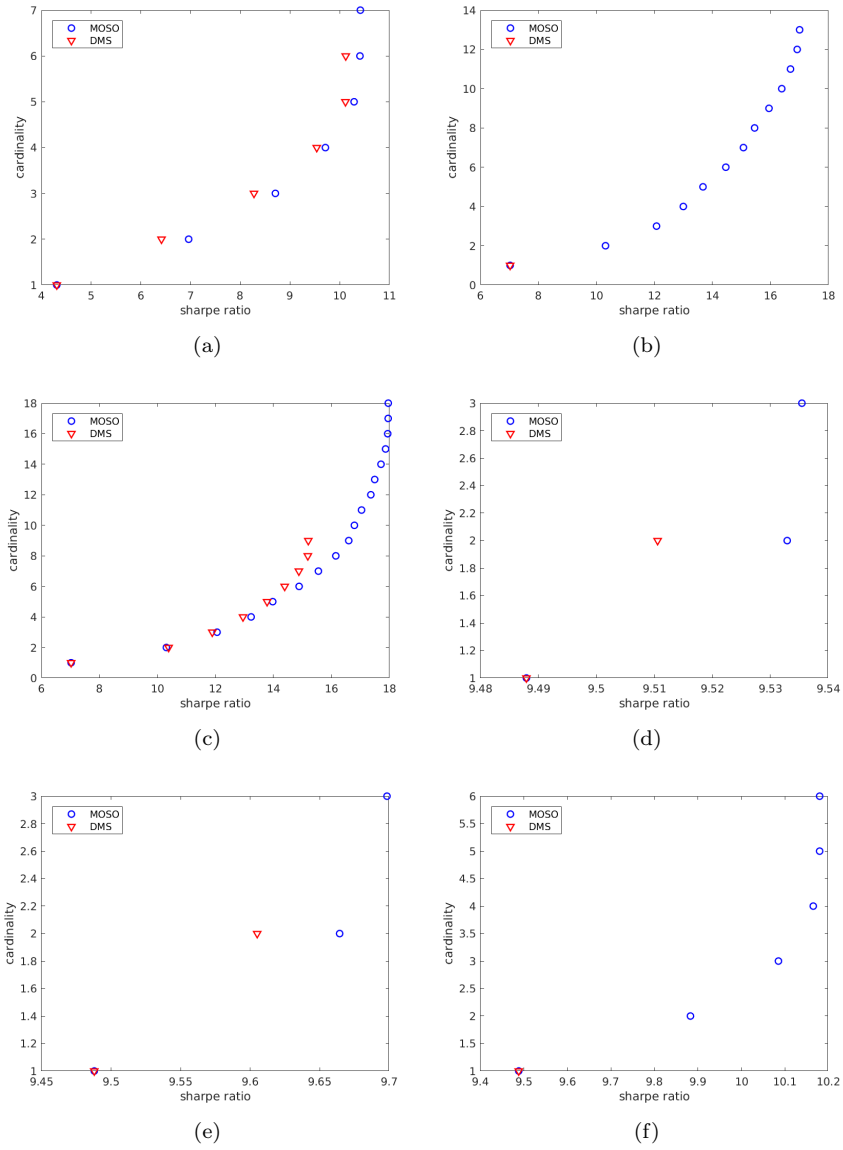

Figure 3.1: Pareto fronts for datasets DTS1 (a), DTS2 (b), DTS3 (c), FF10 (d), FF17 (e) and FF48 (f) starting from a singleton.

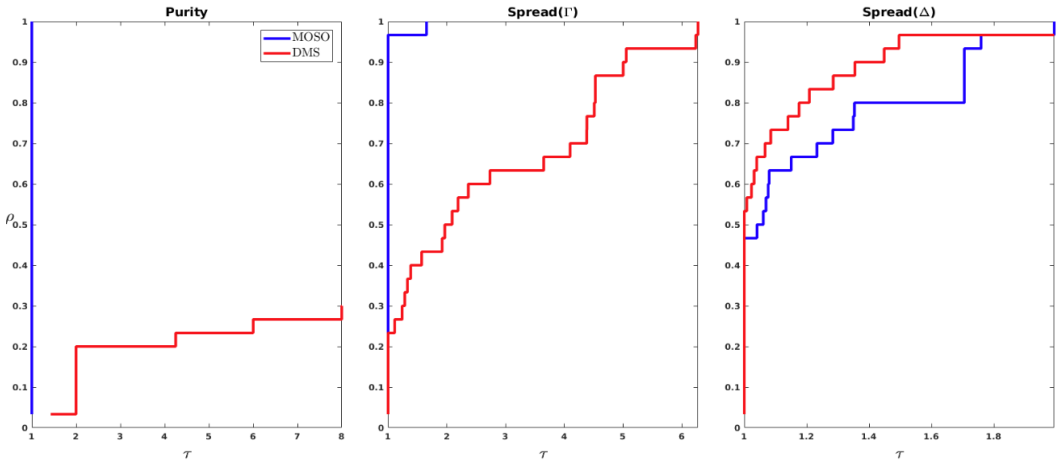

Figure 3.2: Performance profiles computed overall the 30 instances starting from a random list of points.

Chapter 4

An implicit filtering algorithm for derivative-free multiobjective optimization

In this chapter, a MOO problem with *black-box* type functions is considered. Since first and higher order information cannot be computed, a multiobjective adaptation of the derivative-free Implicit Filtering algorithm [36] is proposed. Convergence properties to Pareto-stationary points for the algorithm are stated. Then, in order to compute a Pareto front approximation, also an *a posteriori* version of the algorithm is proposed. Numerical results obtained comparing our framework with *state-of-the-art* derivative-free solvers show the effectiveness of the proposed approach.

4.1 State-of-the-art methods for derivative-free multiobjective optimization

In this section, a list of the most important algorithms for derivative-free multiobjective optimization is reported in order to highlight their relevant features.

BIMADS [6]: it is a bi-objective version of Mesh-Adaptive-Direct-Search algorithm, namely `MADS` [5], for non-smooth multiobjective optimization under general constraints. Basically, `MADS` algorithm for single-objective optimization is an iterative method which tries to improve the current solu-

tion evaluating trial points on a mesh. When an iteration fails to improve the current solution, then the next iteration will evaluate points on a finer mesh.

For what concerns BIMADS, each iteration is organized into three phases: in the first phase, each objective is separately minimized using MADS algorithm, generating an initial set of non-dominated solutions. In the second phase, this set is sorted ascending with respect to one of the two objectives (this results in a descending sort with respect to the other objective), for selecting a "reference point" with the largest gap in the objective space. Then a single-objective optimization problem, which aims to find new non-dominated solutions near the reference point, is solved using MADS. Then, in the last phase, dominated solutions are removed from the current list.

MULTIMADS [7]: a more general version, namely MULTIMADS, is proposed in order to deal with problems with more than two objectives. As in BIMADS, each iteration of MULTIMADS algorithm is organized in three phases, but the most relevant difference between them is about the selection of the reference point in the second phase. In MULTIMADS, an alternate formulation for selecting reference points in $\mathbb{R}^m$, $m > 2$, is proposed.

DMS [12]: Direct-Multisearch algorithm (DMS) is addressed to multiobjective optimization problems with nonlinear constraints which are handled by an extreme barrier function approach. It is an iterative algorithm which generates a sequence of lists of non-dominated solutions.

Each iteration of the algorithm is organized in four different phases. In the first phase a point is selected from the current list according to some rules (e.g. the point which maximizes a Spread metric in the objective space). Then a search-step is performed around the current point through a neighbourhood evaluation. Then the poll-step exploits a positive spanning set (defined by a step size) of a set of directions. In both search-step and poll-step the current list could be enriched with new non-dominated solutions. In the fourth phase, if the current list was never updated during the previous phases, the step size related to the selected point decreases.

DFMO [39]: the Derivative-Free-Multiobjective-Optimization (DFMO) algorithm, as DMS, generates a sequence of sets of non-dominated solutions. For each iteration, each point of the current list is exploited in order to improve the list. A line search technique able to deal with multiple objectives is proposed with a penalty function for handling nonlinear constraints.

4.2 Implicit Filtering algorithm for unconstrained single-objective optimization

In this section we briefly recall the main concepts of the Implicit Filtering algorithm for derivative-free optimization (see [36], and the references therein, for a more thorough description of the algorithm). To this aim and limited to this section, let us consider the following optimization problem:

$$\min_{x \in \mathbb{R}^n} f(x) \tag{4.1}$$

where $f : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable.

In order to introduce the Implicit Filtering algorithm for the solution of Problem (4.1), which is basically an improvement of the well-known coordinate search algorithm, we need to recall some definitions.

Definition 8 (Stencil)**.** *Given a point $x \in \mathbb{R}^n$ and a stepsize $h > 0$ a stencil is a set of points defined as:*

$$S(x, h) := \{x + he_1, \ldots, x + he_n, x - he_1, \ldots, x - he_n\} \tag{4.2}$$

With respect to the coordinate direction e_i, $x + he_i$ and $x - he_i$ can be used to approximate the partial derivative $\frac{\partial f(x)}{\partial x_i}$. Hence, the $S(x, h)$ can be used to compute an approximation of the gradient $\nabla f(x)$.

Definition 9 (Approximated gradient)**.** *Given a point x and a stepsize $h > 0$, the approximated gradient $\nabla_h f(x)$ is defined as follows:*

$$\nabla_h f(x) = \begin{bmatrix} \frac{\partial_h f(x)}{\partial x_1} \\ \vdots \\ \frac{\partial_h f(x)}{\partial x_n} \end{bmatrix} \tag{4.3}$$

where:

$$\frac{\partial_h f(x)}{\partial x_i} = \frac{f(x + he_i) - f(x - he_i)}{2h} \tag{4.4}$$

It is worth noting that, since f is continuously differentiable, $\nabla_h f(x)$ is continuous for every $h > 0$. Once the notion of approximated gradient is stated, the following notion of h-stationarity is given.

Definition 10 (h-stationarity)**.** *Let $x \in \mathbb{R}^n$ and let $\nabla_h f(x)$ its approximated gradient. Then x is a h-stationary point if*

$$||\nabla_h f(x)|| = 0. \tag{4.5}$$

Then, let us recall the definition of a *stencil failure*.

Definition 11 (Stencil failure)**.** *Given a point $x \in \mathbb{R}^n$ and a stepsize $h > 0$ a stencil failure occurs when:*

$$f(x) \leq f(y), \ \forall y \in S(x, h) \tag{4.6}$$

i.e. each point belonging to the stencil cannot improve the objective function with respect to $f(x)$.

Now, we are ready to introduce the sketch of a particular instance of the well-known Implicit Filtering algorithm (see [36] for more general algorithms) for single-objective optimization, which basically consists of two nested loops. The external loop (see Algorithm 6) produces a sequence of stepsizes $\{h_k\}$, such that $h_k \to 0$ for $k \to \infty$, and a sequence of iterates $\{x_k\}$; each x_k is obtained by performing a single Implicit Filtering step, `imstep`, with fixed stepsize h_k. The inner loop (see Algorithm 7) performs a minimization of the objective function f starting from the current iterate $x_k = z_0$. At every iteration, first a stencil is built around z_j. If a stencil failure is not detected, i.e. at least one of the stencil points $y \in S(z_j, h_k)$ strictly decreases the objective function, the approximated gradient is computed. Then, a line search is carried out to obtain a new iterate z_{j+1}, provided that the current point is not h_k-stationary. We note that, in Algorithm 7, when the line search is performed, the "approximated" steepest descent direction is used. In case of failure of the line search, z_{j+1} is set equal to y. The algorithm keeps iterating through the inner cycle until either a stencil failure occurs, or the current point z_j is deemed h_k-stationary.

For a thorough convergence analysis of Algorithm 6, as well as for less basic versions of the Implicit Filtering method, we refer the interested reader to [36].

4.3 Implicit Filtering algorithm for multiobjective optimization

Let us suppose to have the following multiobjective optimization problem with box constraints:

$$\begin{aligned} &\min_{x \in \mathbb{R}^n} (f_1(x), \ldots, f_m(x)) \\ &\ell_i \leq x_i \leq u_i, \ \forall i \in \{1, \ldots, n\} \end{aligned} \tag{4.7}$$

Algorithm 6: `Implicit_Filtering`

input: $x_0 \in \mathbb{R}^n$, $h_0 > 0$, $\gamma, \delta, \tau \in (0,1)$.
ouput: a stationary point x^*.
for $k = 0, 1, \ldots,$ **do**
    $x_{k+1} \leftarrow$ `imstep`(x_k, h_k, τ, γ)
    $h_{k+1} \leftarrow \delta h_k$
end
$x^* \leftarrow x_k$
return x^*

Let us define the feasible set $\mathcal{F}$ of problem (4.7) as:

$$\mathcal{F} := \{x \in \mathbb{R}^n : \ell_i \leq x_i \leq u_i,\ i = 1, \ldots, n\} \tag{4.8}$$

Our main goal is to define an Implicit Filtering-type method to solve the multiobjective optimization problem (4.7). Indeed, we are able to prove that the proposed method converges to Pareto-stationary points of (4.7). In order to export the Implicit Filtering algorithm to a multiobjective context, at least four critical aspects must be carefully considered which are:

- a new definition of stencil failure
- handling a *quasi*-Pareto-stationarity
- approximation of the multiobjective steepest descent direction
- a multiobjective line search

Moreover, bound constraints on the variables must be taken into account in the definition of stencil and approximated gradients

Definition 12 (Constrained approximated gradient)**.** *Given $x \in \mathcal{F}$, $h > 0$, the stencil $S(x, h)$, and an index $i \in \{1, \ldots, m\}$, the approximated gradient with box constraints $\nabla_h f_i$ of f_i at x is defined by*

$$\nabla_h f_i(x) \stackrel{\triangle}{=} \left(\frac{\partial_h f_i(x)}{\partial x_1}, \ldots, \frac{\partial_h f_i(x)}{\partial x_n} \right)^\top$$

Algorithm 7: `imstep`

input: $z_0 \in \mathbb{R}^n$, $h > 0$,$\tau > 0$ $\gamma \in (0,1)$.
ouput: a stationary point z^*.
for $j = 0, 1, \ldots$ **do**
  Build $S(z_j, h)$ and define $\mathcal{S} = \{y \in S(z_j, h) : f(y) < f(z_j)\}$
  if $\mathcal{S} \neq \emptyset$ **then**
    Choose $y \in \mathcal{S}$
  else
    return z_j `// stencil failure: reduce the stepsize`
  end
  Compute $\nabla_h f(z_j)$
  if $||\nabla_h f(z_j)|| \leq \tau h$ **then**
    return z_j `//` h`-stationarity: reduce the stepsize`
  end
  if $\alpha > 0$ *exists s.t.* $f(z_j - \alpha \nabla_h f(z_j)) \leq f(z_j) - \gamma\alpha||\nabla_h f(z_j)||^2$ **then**
    $z_{j+1} \leftarrow z_j - \alpha \nabla_h f(z_j)$
  else
    Set $z_{j+1} \leftarrow y$
  end
end

where, for all $j = 1, \ldots, n$,

$$\frac{\partial_h f_i(x)}{\partial x_j} = \begin{cases} \frac{f_i(x+he_j)-f_i(x-he_j)}{2h}, & x + he_j \in \mathcal{F},\ x - he_j \in \mathcal{F}; \\ \frac{f_i(x+he_j)-f_i(x)}{h}, & x + he_j \in \mathcal{F},\ x - he_j \notin \mathcal{F}; \\ \frac{f_i(x)-f_i(x-he_j)}{h}, & x + he_j \notin \mathcal{F},\ x - he_j \in \mathcal{F}; \\ \infty, & x + he_j \notin \mathcal{F},\ x - he_j \notin \mathcal{F}. \end{cases} \tag{4.9}$$

Note that, $\frac{\partial_h f_i(x)}{\partial x_j} = \infty$ whenever $x + he_j \notin \mathcal{F},\ x - he_j \notin \mathcal{F}$, but this never occurs for sufficiently small values of h; we clarify this point in Lemma 4 below.

Now, we introduce the definition of a *stencil failure* in the multiobjective case and of Pareto h-stationarity.

Definition 13 (Multiobjective stencil failure)**.** *Given $x \in \mathcal{F}$ and a stepsize $h > 0$, a stencil failure at x occurs when there is no $y \in S(x,h) \cap \mathcal{F}$ that dominates x, i.e.*

$$\nexists y \in S(x,h) \cap \mathcal{F} : \quad F(y) \leq F(x).$$

Definition 14 (Pareto h-stationarity)**.** *Given a stepsize $h > 0$, a point $x^* \in \mathcal{F}$ is Pareto h-stationary if, for all $y \in \mathcal{F}$, an index $j \in \{1, \ldots, m\}$ exists, possibly depending on y, such that*

$$\nabla_h f_j(x^*)^T (y - x^*) \geq 0.$$

Reasoning as in Section 1.2, Pareto h-stationarity can be characterized by extending definitions (1.18)-(1.19) for $y(x)$ and $\theta(x)$ to the case of approximated gradients, as follows:

$$\theta(x,h) = \min_{y \in \mathcal{F}} \max_{i=1,\ldots,m} \nabla_h f_i(x)^\top (y - x) \tag{4.10}$$

$$y(x,h) = \arg\min_{y \in \mathcal{F}} \max_{i=1,\ldots,m} \nabla_h f_i(x)^\top (y - x). \tag{4.11}$$

Further, we approximate the steepest descent direction $v(x)$ by

$$v(x,h) = y(x,h) - x.$$

For what concerns the above definitions, we remark that problem (4.11) is a finite min max problem with linear component functions. It can thus be trivially restated as the following linear programming (LP) problem

$$\begin{aligned} &\min_{y,\beta} \ \beta \\ &\nabla_h f_i(x)^\top (y - x) \leq \beta, \quad i = 1, \ldots, m, \\ &y \in \mathcal{F} \end{aligned} \tag{4.12}$$

It can be easily seen (see point (i) of Proposition 10 in the Appendix) that $\theta(x,h) \leq 0$ for all $x \in \mathcal{F}$ and $h > 0$, so that x^* is Pareto h-stationary if and only if $\theta(x^*, h) = 0$. Furthermore, (see point (ii) of Proposition 10 in the Appendix) for any given $x \in \mathcal{F}$, it results

$$\lim_{h \to 0^+} \theta(x,h) = \theta(x).$$

Now, we develop a first version of an Implicit Filtering algorithm which converges asymptotically to a single Pareto-stationary point. As in the Implicit Filtering framework, our algorithm consists of two nested loops. The

Algorithm 8: `MultiObjectiveImplicitFiltering (MOIF)`

input: $x_0 \in \mathcal{F}$, $h_0 > 0$, $\gamma, \delta, \tau \in (0, 1)$
output: an optimal point x^*
for $k = 0, 1, 2, \ldots$ **do**
    $x_{k+1} \leftarrow$ `imstepMulti`(x_k, h_k, τ, γ)
    $h_{k+1} \leftarrow \delta h_k$
end
$x^* \leftarrow x_k$
return x^*

outermost loop, which is reported in Algorithm 8, is very similar to the outer loop of Algorithm 6 for single-objective optimization reported in Section 4.2.

The innermost loop is the most important one and is reported in the following algorithm `imstepMulti`.

Algorithm 9: `imstepMulti`

input: $\tilde{z}_0 \in \mathcal{F}, h > 0, \tau > 0, \gamma \in (0,1)$
output: a sub-optimal point z^*
for $j = 0, 1, \ldots$ **do**
    `stencilFailure` $\leftarrow$ FALSE
    $w \leftarrow \tilde{z}_j$
    while not *stencilFailure* **do**
        Build $S(w,h)$
        if $S(w,h) \cap \mathcal{F} = \emptyset$ **then**
            return w `// stencil infeasible: reduce the stepsize`
        else
            $\mathcal{S} \leftarrow \{y \in S(w,h) \cap \mathcal{F} : F(y) \leq F(w) - \mathbf{1}\gamma h\}$
        end
        if $\mathcal{S} \neq \emptyset$ **then**
            Choose $y \in \mathcal{S}$ and set $w = y$
        else
            `stencilFailure` $\leftarrow$ TRUE
        end
    end
    $z_j \leftarrow w$
    if $\exists i \in \{1, \ldots, n\}$ *s.t.* $z_j + he_i, z_j - he_i \notin \mathcal{F}$ **then**
        return z_j `// approximated gradient undetermined: reduce the stepsize`
    end
    $\theta_j \leftarrow \theta(z_j, h)$
    $y_j = y(z_j, h)$
    $v_j = y_j - z_j$
    if $\theta_j \geq -\tau h$ **then**
        return z_j `// Pareto` h`-stationarity: reduce the stepsize`
    end
    $\alpha_j \leftarrow$ `Goldstein`$(z_j, v_j, h, \theta_j, \gamma)$
    if $\alpha_j |\theta_j| \leq \tau h$ **then**
        return z_j `// line search-failure: reduce the stepsize`
    end
    $\tilde{z}_{j+1} \leftarrow z_j + \alpha_j v_j$
end
$z^* \leftarrow z_j$
return z^*

As we can see, the internal loop is somewhat different from Algorithm 7 and is composed by two parts:

- direct search along the coordinate axis;
- (possible) line search along the approximated steepest descent direction (obtained by solving an LP).

The main difference between Algorithms 7 and 9 resides in the behavior of the algorithms in case of a stencil failure. When a stencil failure is detected by Algorithm 7, the inner loop is abandoned so that the step size is reduced. On the contrary, in Algorithm 9, when a stencil failure is detected, the inner loop tries to execute a line search along the approximated steepest descent direction.

We decided to modify the basic Implicit Filtering behavior by drawing inspiration from the recently proposed algorithm `DMS` (see Remark 5 below). We move along coordinate axis by a fixed step h. At iteration j of Algorithm `imstepMulti`, we have a stencil failure when no feasible stencil point $y \in S(w,h) \cap \mathcal{F}$ exists such that

$$F(y) \leq F(w) - \mathbf{1}\gamma h.$$

When a stencil failure finally occurs, we proceed trying to find a descent direction using $\nabla_h F(z_j)$. In particular, θ_j and $v_j = y_j - z_j$ are computed by solving a linear programming subproblem of type (4.12) with exact gradients replaced by approximated ones. In this respect, we note that this calculation does not increase the number of objective function evaluations, because the approximated gradient is computed through stencil points already computed.

If we find a sufficiently good feasible direction v_j, i.e. such that $\theta(z_j, h) < -\tau h$, we perform a line search along the computed direction v_j. We recall that direction v_j is such that the unitary stepsize is feasible, i.e. $z_j + v_j \in \mathcal{F}$ and v_j is a feasible direction.

Remark 2. *If $\theta(z_j, h) \geq -\tau h$, the current point z_j is deemed h-stationary and the current stepsize h is reduced by the algorithm. This could also mean that the current gradient approximation is not suitable. Further, even if $\theta(z_j, h) < -\tau h < 0$, there is no guarantee that direction v_j is a descent direction for the objective functions.*

Remark 3. *Direction v_j is computed by using approximations of the objective functions gradients. Hence, for a fixed stepsize h, it might well not be a descent direction at the current iterate z_j.*

The above two observations suggested us to use a Goldstein line search (in place of an Armijo one) with initial stepsize α_j equal to h, i.e. the stepsize used in the algorithm to build the stencil and compute the approximated objective functions gradients. In this way, we are able to immediately verify the quality of the direction, thus avoiding a potentially large number of function evaluations.

Algorithm 10: `Goldstein`

input: $x \in \mathbb{R}^n, v \in \mathbb{R}^n, h > 0, \theta < 0, \gamma \in (0,1)$
output: a stepsize $\alpha \geq 0$
if $F(x + hv) \leq F(x) + \mathbf{1}\gamma(h\theta)$ **and** $x + hv \in \mathcal{F}$ **then**
    $\beta \leftarrow 0$
    while $x + 2^{\beta+1}hv \in \mathcal{F}$ **and** $F(x + 2^{\beta+1}hv) \leq F(x) + \mathbf{1}\gamma(2^{\beta+1}h\theta)$ **do**
        $\beta \leftarrow \beta + 1$
    end
    $\alpha \leftarrow 2^{\beta}h$
else
    $\alpha \leftarrow 0$
end
return α

Remark 4. *We observe that, whenever the test $\alpha_j|\theta_j| \leq \tau h$ holds, as $|\theta_j| > \tau h$, we have that the produced α_j is "sufficiently small", so that, the current point can be considered an approximation of a Pareto h-stationary point and, as a consequence, the finite-difference stepsize h is reduced.*

Remark 5. *As we said before, the direct search along the coordinate directions draws inspiration from `DMS`, which is an algorithmic framework that extends the class of direct search methods for single optimization to nonsmooth, multiobjective optimization. In the general setting of `DMS`, the objective functions are sampled along suitable sets of search directions, the acceptance criterion is based on the Pareto dominance, and a list of feasible non-dominated points is updated. The simplest strategy of the framework is that of considering lists formed by a single point and to use the set of coordinate directions as search directions. This strategy is similar to that defined in the direct search block of our algorithm. However, in our framework the Im-*

plicit Filtering phase generates an approximated "steepest descent direction" for all objective functions, by solving the min max *problem (see (4.11)), and this is the key ingredient, coupled with the line search, to ensure convergence properties.*

4.4 Convergence analysis

First of all we prove that Algorithm `imstepMulti` is well defined. To this aim, in the following lemma we show that for every $j \geq 0$ and $\tilde{z}_j$ a point z_j is produced, i.e. `stencilFailure` is eventually set to $TRUE$.

Lemma 1. *With reference to Algorithm `imstepMulti`, for every $j \geq 0$, after a finite number of iterations of the while-loop, either stencil infeasibility is detected or stencil failure is obtained, i.e. `stencilFailure` is eventually set to $TRUE$.*

Proof. Let us assume by contradiction that for a given iteration j the while-loop never terminates, i.e. the stencil is always feasible and `stencil-Failure` is never set to TRUE. In this case, let us denote by w_t the stencil center point at the generic t-th iteration of the while loop, and by w_{t+1} the point of the next stencil. Of course, for every t it results

i) $w_t \in \mathcal{F}$;

ii) $F(w_{t+1}) \leq F(w_t) - \mathbf{1}\gamma h$.

Recalling that $w_0 = \tilde{z}_j$ and point ii) above, we can write

$$F(w_{t+1}) \leq F(w_t) - \mathbf{1}\gamma h \leq F(w_{t-1}) - \mathbf{1}(2\gamma h) \leq \cdots \leq F(\tilde{z}_j) - \mathbf{1}(t+1)\gamma h.$$

Taking the limit for $t \to \infty$ in the above relation we would obtain

$$\lim_{t\to\infty} f_i(w_t) = -\infty, \quad i = 1, \ldots, m,$$

which is a contradiction with the continuity of F, compactness of $\mathcal{F}$, and $\{w_t\} \subset \mathcal{F}$. □

By virtue of Lemma 1,

- either $z_j = \tilde{z}_j$ and $F(z_j) = F(\tilde{z}_j)$
- or z_j is such that $F(z_j) \leq F(\tilde{z}_j) - \mathbf{1}\gamma h$.

Then, we prove that the Goldstein procedure is well-defined, namely that the while-loop in the Goldstein procedure cannot infinitely cycle.

Lemma 2. *For every $x \in \mathcal{F}$, $h > 0$, $\theta < 0$, and $v \in \mathbb{R}^n$, the Goldstein procedure always returns a value $\alpha \geq 0$.*

Proof. If $F(x + hv) \not\leqq F(x) + \mathbf{1}\gamma(h\theta)$ or $x + hv \notin \mathcal{F}$, the procedure immediately returns $\alpha = 0$.

Hence, we have only to analyze the case when $F(x+hv) \leq F(x)+\mathbf{1}\gamma(h\theta)$ and $x + hv \in \mathcal{F}$, and show that the while-loop cannot infinitely cycle. Let us proceed by contradiction and assume that the while-loop infinitely cycle. This means that, for every $\beta = 0, 1, 2, \ldots$, we have

$$x + 2^{\beta+1}hv \in \mathcal{F} \text{ and } F(x + 2^{\beta+1}hv) \leq F(x) + \mathbf{1}\gamma(2^{\beta+1}h\theta).$$

For β sufficiently large, this is a contradiction with the compactness of $\mathcal{F}$ and continuity of F, and concludes the proof. □

Now, concerning the convergence of Algorithm `MOIF` (the `Multi Objective Implicit Filtering` procedure described in Algorithm 8), the first thing we need to prove is that the stepsize h_k converges to zero. Thus, we state the following lemma.

Lemma 3. *Algorithm `MOIF` generates infinite sequences $\{x_k\} \subset \mathcal{F}$ and $\{h_k\} \subset \mathbb{R}^+$, such that*

$$\lim_{k\to\infty} h_k = 0. \tag{4.13}$$

Proof. We assume by contradiction that for an index $\bar{k}$ Algorithm `imstepMulti` does not terminate, thus producing infinite sequences $\{\tilde{z}_j\}$, $\{z_j\}$, $\{\alpha_j\}$, $\{\theta_j\}$, $\{v_j\}$, and $\{f_i(z_j)\}$, for $i = 1, \ldots, m$. Let $\bar{h} = h_{\bar{k}}$. Furthermore, for all $j = 0, 1, \ldots,$

i) $\theta_j = \theta(z_j, \bar{h}) < -\tau\bar{h} < 0$, and

ii) $z_j + \alpha_j v_j \in \mathcal{F}$ and $\alpha_j|\theta_j| > \tau\bar{h}$.

Hence, for all $i \in \{1, \ldots, m\}$ and $j \geq 0$, we have

$$f_i(z_{j+1}) \leq f_i(\tilde{z}_{j+1}) = f_i(z_j + \alpha_j v_j) \leq f_i(z_j) + \gamma\alpha_j\theta_j,$$

that is

$$f_i(z_j) - f_i(z_{j+1}) \geq -\gamma\alpha_j\theta_j. \tag{4.14}$$

Since $\{f_i(z_j)\}$, for $i = 1, \ldots, m$, are non-increasing sequences, and $z_j \in \mathcal{F}$ for all j, then the continuity of f_i ensures that

$$\lim_{j\to\infty} f_i(z_j) - f_i(z_{j+1}) = 0, \ \ \forall i = 1, \ldots, m.$$

Above limits and relation (4.14) imply that

$$\lim_{j\to\infty} \alpha_j |\theta_j| = 0.$$

Thus, for j sufficiently large we will have $\alpha_j|\theta_j| \leq \tau\bar{h}$. In other words, the test after the Goldstein procedure will be satisfied, and hence Algorithm `imstepMulti` will terminate, within a finite number of inner iterations, in contradiction with our initial assumption. Then (4.13) is easily proved, as $\{h_k\}$ is an infinite sequence such that $h_{k+1} = \delta h_k$ with $\delta \in (0, 1)$. □

Before stating the main theorem, we prove the following result.

Lemma 4. *Let $\{x_k\}_{k\in K}$ be a subsequence produced by Algorithm `MOIF` and assume that $x_k \to \bar{x}$ for $k \in K$ and $k \to \infty$. Then, for $k \in K$ and k sufficiently large we have that for $j = 1, \ldots, n$ at least one of the following conditions holds*

$$x_k + h_k e_j \in \mathcal{F}$$

$$x_k - h_k e_j \in \mathcal{F}.$$

Proof. The points of the subsequence $\{x_k\}_{k\in K}$ belong to the closed set $\mathcal{F}$, so that, $\bar{x} \in \mathcal{F}$. Since $\mathcal{F}$ is defined by box constraints, for $j = 1, \ldots n$ we have that at least one of the vectors e_j or $-e_j$ is a feasible direction at $\bar{x}$. Let $d_j \in \{e_j, -e_j\}$ be a feasible direction at $\bar{x}$, and let $\mathcal{D}(\bar{x})$ be the set of feasible directions at $\bar{x}$. Again, since $\mathcal{F}$ is defined by linear constraints, from known results (see, e.g., Proposition 4 in [40] and Proposition A1 in [38]) it follows:

(i) $\mathcal{D}(\bar{x}) \subseteq \mathcal{D}(x_k)$ for $k \in K$ and k sufficiently large;

(ii) given $d \in \mathcal{D}(\bar{x})$, there exists $\bar{t} > 0$ such that, for all $t \in (0, \bar{t}]$, we have $x_k + td \in \mathcal{F}$, for $k \in K$ and k sufficiently large.

Then for any $j = 1, \ldots, n$, being $d_j \in \{e_j, -e_j\}$ a feasible direction at $\bar{x}$ and recalling that $h_k \to 0$ for $k \in K$ and $k \to \infty$, it follows from (ii) that

$$x_k + h_k d_j \in \mathcal{F}$$

for $k \in K$ and k sufficiently large, and the thesis is proved. □

Finally, we can state the main theorem concerning the convergence of Algorithm `MOIF`.

Theorem 1. *Let $\{x_k\}$ be a sequence produced by Algorithm `MOIF`. Then $\{x_k\}$ admits limit points, and each one of them is Pareto-stationary for Problem (4.7).*

Proof. By construction, $\{x_k\} \subset \mathcal{F}$. Hence, by compactness of $\mathcal{F}$ and using Lemma 3, there exists a limit point $\bar{x} \in \mathcal{F}$ of $\{x_k\}$ and an infinite subset of iteration indices $K \subset \{0, 1, 2 \dots\}$ such that

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

and

$$\lim_{k \in K, k \to \infty} h_k = 0. \tag{4.15}$$

From Lemma 4 it follows that, for $k \in K$ and sufficiently large, procedure `imstepMulti`(x_k, h_k, τ, γ) can never return because $S(x_k, h_k) \cap \mathcal{F} = \emptyset$ or because $x_k + h_k e_i \notin \mathcal{F}$ and $x_k - h_k e_i \notin \mathcal{F}$, for all $i = 1, \dots, n$, i.e. the stencil is feasible and all the components of the approximated gradient at x_k are finite as defined by (4.9).

Then, after relabelling (if necessary) the index set K, we can split K into two subsets, K_1 and K_2, defined as

$$\begin{aligned} K_1 &= \{k \in K : -\tau h_k \leq \theta(x_k, h_k) \leq 0\} \text{ and} \\ K_2 &= \{k \in K \setminus K_1 : \ \alpha_k |\theta(x_k, h_k)| \leq \tau h_k\}. \end{aligned}$$

Note that, since K is infinite, K_1 and K_2 cannot be both finite.

First, let us suppose that K_1 is infinite. From the definition of K_1, using assertion (ii) of Proposition 10 in the Appendix and (4.15), we obtain that

$$\lim_{k \to \infty, k \in K_1} \theta(x_k, h_k) = \theta(\bar{x}) = 0,$$

which completes the proof in this case.

Let us now suppose that K_2 is infinite. We proceed by contradiction and assume that $\bar{x}$ is not Pareto-stationary, i.e.

$$\theta(\bar{x}) < 0. \tag{4.16}$$

Then, from the definition of K_2 and considering (4.15), we get

$$\lim_{k\in K_2, k\to\infty} \alpha_k |\theta(x_k, h_k)| = 0,$$

which, recalling (ii) of Proposition 10 in the Appendix, (4.15) and (4.16), yields

$$\lim_{k\in K_2, k\to\infty} \alpha_k = 0. \tag{4.17}$$

Furthermore, it must exist an infinite subset $\bar{K}_2 \subset K_2$ such that one of the following two cases can occur:

i) either $\alpha_k = 0$ for $k \in \bar{K}_2$

ii) or $\alpha_k = 2^{\beta_k} h_k > 0$ for $k \in \bar{K}_2$.

Point i). In this case, recalling that $v(x_k, h_k) = y(x_k, h_k) - x_k$ so that $x_k + v(x_k, h_k) = y(x_k, h_k) \in \mathcal{F}$ by definition, that for k sufficiently large $h_k < 1$, and that $\mathcal{F}$ is a convex set, we have that $x_k + h_k v_k \in \mathcal{F}$ for k sufficiently large and, by definition of the Goldstein procedure,

$$F(x_k + h_k v_k) \not\leq F(x_k) + \mathbf{1}\gamma h_k \theta_k, \tag{4.18}$$

where we introduced the notations $v_k = v(x_k, h_k)$ and $\theta_k = \theta(x_k, h_k)$. If (4.18) holds, we can write, for at least an index $\ell \in \{1, \ldots, m\}$ (which can depend on k),

$$\frac{f_\ell(x_k) - f_\ell(x_k + h_k v_k)}{h_k} < -\gamma\theta_k \leq -\gamma \nabla_{h_k} f_\ell(x_k)^\top v_k.$$

Then, by the Mean Value Theorem, an $\tilde{h}_k \in (0, h_k)$ exists such that

$$-\nabla f_\ell(x_k + \tilde{h}_k v_k)^\top v_k < -\gamma\theta_k \leq -\gamma \nabla_{h_k} f_\ell(x_k)^\top v_k. \tag{4.19}$$

Moreover,

$$\begin{aligned} -\gamma \nabla_{h_k} f_\ell(x_k)^\top v_k &= \gamma(\nabla f_\ell(x_k) - \nabla_{h_k} f_\ell(x_k))^\top v_k - \gamma \nabla f_\ell(x_k)^\top v_k \\ &\leq \gamma \|\nabla f_\ell(x_k) - \nabla_{h_k} f_\ell(x_k)\| \|v_k\| - \gamma \nabla f_\ell(x_k)^\top v_k. \end{aligned}$$

Now, using (4.19) and the above relation, and recalling that the sequence of search directions $\{v_k\}$ is bounded, a fixed $\bar{\ell} \in \{1, \ldots, m\}$, a direction $\bar{v} \in \mathbb{R}^n$, and an infinite subset $K_3 \subseteq \bar{K}_2$ exist such that $\forall k \in K_3$

$$\begin{aligned} &-\nabla f_{\bar{\ell}}(x_k + \tilde{h}_k v_k)^\top v_k < \\ &\qquad -\gamma\theta_k \leq \gamma \|\nabla f_{\bar{\ell}}(x_k) - \nabla_{h_k} f_{\bar{\ell}}(x_k)\| \|v_k\| - \gamma \nabla f_{\bar{\ell}}(x_k)^\top v_k \end{aligned} \tag{4.20}$$

and

$$\lim_{k \in K_3, k \to \infty} v_k = \bar{v}.$$

Taking the limit in (4.20) and recalling Proposition 9 and Proposition 10 in Appendix, we then obtain

$$-\nabla f_{\bar{\ell}}(\bar{x})^\top \bar{v} \ \leq -\gamma\theta(\bar{x}) \leq -\gamma \nabla f_{\bar{\ell}}(\bar{x})^\top \bar{v},$$

from which $(1-\gamma)\nabla f_{\bar{\ell}}(\bar{x})^\top \bar{v} \geq 0$ and $\theta(\bar{x}) \geq \nabla f_{\bar{\ell}}(\bar{x})^\top \bar{v}$ follow. Together with $\gamma \in (0,1)$ these yield $0 > \theta(\bar{x}) \geq \nabla f_{\bar{\ell}}(\bar{x})^\top \bar{v} \geq 0$, i.e. a contradiction.

Point ii). In this case, $\alpha_k = 2^{\beta_k} h_k > 0$. From (4.17), recalling that v_k is computed in such a way that $x_k + v_k \in \mathcal{F}$, see e.g. (4.11), for $k \in \bar{K}_2$ and k sufficiently large we have $x_k + 2\alpha_k v_k \in \mathcal{F}$. Hence, the Goldstein procedure returns α_k such that, recalling (4.17) and the convexity of $\mathcal{F}$,

$$x_k + 2\alpha_k v_k \in \mathcal{F} \quad \text{and} \quad F(x_k + 2\alpha_k v_k) \not\leq F(x_k) + \mathbf{1}\gamma(2\alpha_k \theta_k). \tag{4.21}$$

Then, reasoning as above, we can again conclude that $\bar{x}$ is Pareto-stationary for Problem (4.7), thus raising yet a contradiction and concluding the proof. □

4.5 Computational experiments

In this section we report the numerical results of experiments performed in order to assess the effectiveness and the efficiency of the proposed algorithm, both when it is used to generate a single non-dominated point (i.e. when preferences of the decision maker are disregarded), and to generate a set of non-dominated solutions (i.e. when preferences of the decision maker are taken into account *a posteriori*). In both situations, our strategy is compared with the Direct Multisearch algorithm (`DMS`) proposed in [12]. We acknowledge that algorithm `DFMO` proposed in [39] is also suitable for the solution of Problem (4.7) when derivatives are not available. However, considering the results reported in [39] and in particular Figure 2 therein, `DMS` appears to be better than `DFMO`, at least when the coordinate directions are used as search directions in both algorithms.

Test problems. We considered the set of 100 multiobjective problems used in [12], whose dimension n is in the range $[1, 30]$ and with a number m of objectives belonging to the set $\{2, 3, 4\}$.

Implementation details. We have implemented Algorithm 8 in MATLAB© and we tested it on an Intel 1.2 GHz quad-core multithread with 8GB RAM. Parameters of algorithm `MOIF` have been set as follows

$$h_0 = 1, \quad \tau = 10^{-2}$$
$$\delta = 0.5, \quad \gamma = 10^{-5}.$$

The computation of $\theta(x, h)$ and $y(x, h)$, as defined in (4.10) and (4.11), respectively, in Algorithm `MOIF` is carried out by using the `linprog` function of MATLAB©.

In the experiments, `MOIF` is stopped whenever either $h_k \leq 10^{-3}$ or the number of function evaluations exceeds 20,000. Furthermore, as concerns the implementation of algorithm `imstepMulti`, i.e. Algorithm 9, the instruction "Choose $y \in \mathcal{S}$" is realized so that y is the first point in $\mathcal{S}$ not dominated by any other point in $\mathcal{S}$.

As for `DMS`, all of its parameters (except for `Pareto_front`) have been set to their default values. In particular, we have

`stop_alfa = 1`	`tol_stop =` 10^{-3}
`stop_feval = 1`	`max_fevals =` $20,000$

so that the stopping criteria of `DMS` are the same as those used in `MOIF`. Both solvers start from the centroid of the feasible region $\mathcal{F}$, i.e.

$$(x_0)_i = \frac{u_i + \ell_i}{2}, \quad \forall\ i = 1, \ldots, n.$$

4.5.1 Computation of a single non-dominated solution

We compare the performance of our algorithm `MOIF` (i.e. Algorithm 8) with the one of `DMS` when it is used to generate a single non-dominated solution, i.e. when the calling parameter `Pareto_front` is set to 0.

The first thing that we observe is that, in 54 out of 100 problems, the Goldstein line search is never performed since the Pareto $h-$stationarity condition

$$\theta_j \geq -\tau h$$

is always satisfied. When this happens, the two algorithms `MOIF` and `DMS` show the same behaviour and produce the same non-dominated point.

In the remaining 46 problems, at least one line search is performed with success by algorithm `MOIF` during the optimization process. In 20 out of these

46 problems, MOIF determines a point that dominates the point computed by DMS, while the point provided by DMS never happens to dominate the point determined by MOIF. Hence, we can conclude that these results show the good performance of MOIF in terms of quality of the computed solution.

In the 26 problems where the two solvers computed solutions that do not dominate each other, we have the following situation in terms of objective function evaluations. In 5 out of 26 test problems MOIF required a lower number of function evaluations than that required by DMS. However, this situation is not surprising since, as we may expect, the line search procedure of MOIF requires additional function evaluations with respect to the plain coordinate search performed by both solvers.

On the whole, we may conclude that algorithm MOIF, which combines a coordinate search phase with an Implicit Filtering strategy, shows a good ability to produce non-dominated solutions, confirming the viability of the proposed approach. It can also be noted that, due to the additional burden of the Goldstein line search, MOIF might be somewhat more expensive than DMS.

4.5.2 Computation of a set of non-dominated solutions

When an approximation of the Pareto front is required, like e.g. in *a posteriori* methods, drawing inspiration from DMS, we have defined a version of algorithm MOIF aimed at approximating the whole Pareto front which we call $\texttt{MOIF}_{front}$ (see Algorithm 11). In the following, we discuss the main aspects that distinguish $\texttt{MOIF}_{front}$ from MOIF (i.e. the proposed algorithm aimed at computing a single non-dominated solution) and DMS.

The main difference between MOIF and $\texttt{MOIF}_{front}$ is that every iteration of $\texttt{MOIF}_{front}$ is characterized by a set, say L_k, of point-stepsize pairs rather than the single pair (x_k, h_k), as in MOIF and as it is common in algorithms for single-objective optimization.

Management of the sequence of sets $\{L_k\}$ is performed drawing inspiration both from DMS [12] and DFMO [39]. Since this aspect and, in particular, the generation of L_{k+1} starting from L_k is not trivial, we describe it with some detail.

For each k, let L_k be the following finite set

$$L_k = \{(x_i, h_i),\ x_i \in \mathcal{F}, h_i > 0, i = 1, \ldots, r_k\},$$

where $r_k = |L_k|$ and h_i is the (tentative) stepsize associated with point x_i.

At each iteration k, a current pair $(x_i^k, h_i^k) \in L_k$ is selected (according to a certain criterion). Then, let

$$L'_k = \left((S(x_i^k, h_i^k) \cap \mathcal{F}) \times \{h_i^k\} \right) \cup L_k$$

and

$$\tilde{L}_k = \{(x_i, h_i) \in L'_k : \nexists (x_j, h_j) \in L'_k \text{ s.t. } F(x_j) \leq F(x_i)\}. \tag{4.22}$$

Then, the new set L_{k+1} is defined as:

$$L_{k+1} = \tilde{L}_k \qquad \text{when } \tilde{L}_k \neq L_k, \tag{4.23a}$$

$$L_{k+1} = \left\{ (x_i, h_i) \in L_k \cup \{(w_k, \alpha_k)\} : \nexists (x_j, h_j) \in L_k \text{ s.t. } F(x_j) \leq F(x_i) \right\} \qquad \begin{array}{l} \text{when } \tilde{L}_k = L_k, \\ \theta_k < -\tau h_i^k, \\ \alpha_k |\theta_k| > \tau h_i^k \end{array} \tag{4.23b}$$

$$L_{k+1} = L_k \setminus \{(x_i^k, h_i^k)\} \cup \{(x_i^k, \delta h_i^k)\} \qquad \text{otherwise} \tag{4.23c}$$

where

- $\theta_k = \theta(x_i^k, h_i^k)$
- $v_k = v(x_i^k, h_i^k)$
- $w_k = x_i^k + \alpha_k v_k$
- $\alpha_k = \texttt{Goldstein}(x_i^k, v_k, h_i^k, \theta_k, \gamma)$

Note that when the algorithm is not able to find any new non-dominated point, that is, when both the coordinate search does not produce any new non-dominated point (i.e. $\tilde{L}_k = L_k$), and the Goldstein line search either is not performed (because either the approximated gradient is undetermined or $\theta_k \geq -\tau h_i^k$) or produces an unsuitable stepsize (i.e. $\alpha_k |\theta_k| \leq \tau h_i^k$), L_{k+1} is obtained from L_k by replacing the selected pair (x_i^k, h_i^k) by $(x_i^k, \delta h_i^k)$. The same is done in case of infeasible stencil, that is when $L_k = L'_k = \tilde{L}_k$ because $S(x_i^k, h_i^k) \cap \mathcal{F} = \emptyset$.

As we already pointed out in Remark 5, theoretical convergence properties of Algorithm $\texttt{MOIF}_{front}$ (at least under smoothness assumptions) would derive from the Goldstein line search procedure.

However, carrying out such a theoretical convergence analysis for algorithm $\texttt{MOIF}_{front}$ would considerably burden the thesis. Indeed, before proceeding with the theoretical analysis it would be necessary to formally state

Algorithm 11: $\texttt{MOIF}_{front}$

input : $\gamma, \delta, \tau \in (0,1)$, $L_0 = \{(x_i, h_i),\ x_i \in \mathcal{F}, h_i > 0, i = 1, \dots, r_0\}$ initial set of non-dominated points
ouput: an approximation of Pareto front L^*
for $k = 0, 1, \dots$ **do**
  Select $(x_i^k, h_i^k) \in L_k$ and compute $\tilde{L}_k$ as in (4.22)
  if $\tilde{L}_k \neq L_k$ **then**
    $L_{k+1} \leftarrow \tilde{L}_k$
  else
    Compute $\theta_k = \theta(x_i^k, h_i^k)$, $y_k = y(x_i^k, h_i^k)$, $v_k = y_k - x_i^k$
    if $\theta_k \geq -\tau h_i^k$ **then**
      $L_{k+1} \leftarrow L_k \setminus \{(x_i^k, h_i^k)\} \cup \{(x_i^k, \delta h_i^k)\}$
    else
      Compute $\alpha_k = \text{Goldstein}(x_i^k, v_k, h_i^k, \theta_k, \gamma)$
      if $\alpha_k |\theta_k| \leq \tau h_i^k$ **then**
        $L_{k+1} \leftarrow L_k \setminus \{(x_i^k, h_i^k)\} \cup \{(x_i^k, \delta h_i^k)\}$
      else
        Set L_{k+1} as in (4.23b)
      end
    end
  end
end
$L^* \leftarrow L_k$
return L^*

what a sequence of points is supposed to be in an algorithmic framework that generates sequences of sets of points. Most probably, it could be necessary to proceed as in [39] and as in Chapter 2 using "linked sequences of points". Then, stationarity results could be given with reference to (some) linked sequences.

Numerical results. Comparison of $\texttt{MOIF}_{front}$ and DMS in this context is carried out by means of the Purity and Spread (both Γ and Δ) metrics used in [12], and by using performance profiles [17].

We recall that the Purity metric measures the quality of the generated front, i.e. how good the non-dominated points computed by a solver are with

respect to those computed by any other solver. Note that, for each problem p, the "reference" Pareto front F_p^{true} is calculated by first computing

$$\tilde{F}_p^{true} = F_{p,DMS} \cup F_{p,\texttt{MOIF}},$$

where $F_{p,s}$ denotes the set of non-dominated solutions found by solver s for problem p, and then removing from this set any dominated solution as in (1.23).

On the other hand, the Spread metrics are essential to measure the uniformity of the generated front in the objectives space.

In the experiments $\texttt{MOIF}_{front}$ is stopped at iteration k, when the number of function evaluations exceeds 20,000, or when the following stepsize criterion holds:

$$\max_{(x_i,h_i)\in L_k} h_i \leq 10^{-3}. \tag{4.24}$$

As concerns the choice of $x_k \in L_k$, we select the point with the highest Γ value as in [12].

In figure 4.1, comparison between $\texttt{MOIF}_{front}$ and DMS is reported in terms of the above mentioned metrics.

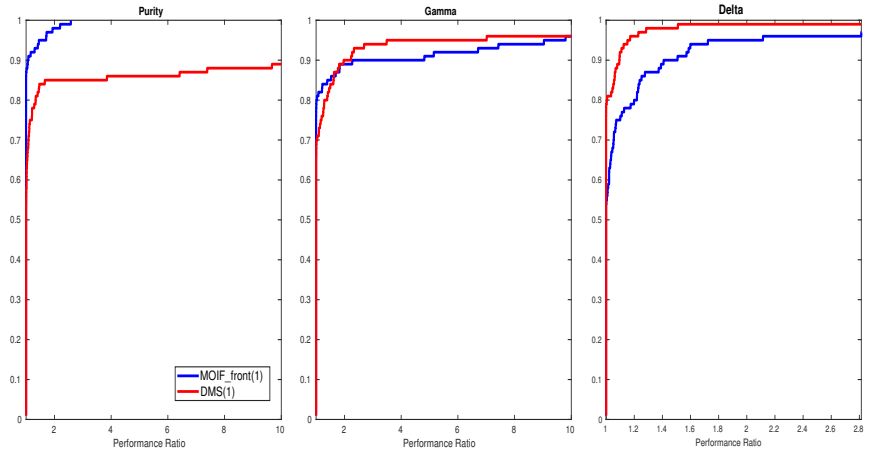

Figure 4.1: Comparison between $\texttt{MOIF}_{front}$ and DMS when both solvers start from the centroid of the feasible region $\mathcal{F}$

As we can see, $\texttt{MOIF}_{front}$ outperforms DMS in terms of Purity, while it can be considered equivalent in terms of Spread Γ. However, DMS outperforms $\texttt{MOIF}_{front}$ in terms of Spread Δ.

As we may expect, the line search phase is very effective when the quality of the generated points is regarded. This is confirmed by the results in terms of Purity. We observe that many points of the current list L_k may be dominated by the points generated by the line search along "good" descent directions. Hence, the cardinality of the list of non-dominated points may tend to quickly reduce with respect to a strategy based on the pure coordinate search as in DMS. This may lead to a generated front with a lower degree of uniformity compared with that generated by DMS.

In order to take into account the effect of the line search in terms of the Spread metrics, we defined a version of the algorithm that uses a suitable condition to decide whether or not to perform the line search at a given iteration. More specifically, let $c_p \in [0, 1]$ be a parameter which we call Purity coefficient. Then, in Algorithm MOIF$_{front}$, given the current pair $(x_i^k, h_i^k) \in L_k$ and θ_k, the Goldstein line search is performed only if

$$\theta_k < -\bar{\tau} h_i^k \quad \text{where} \quad \bar{\tau} = \begin{cases} \tau & \text{if } h_i^k \leq c_p \max_{(x,h)\in L_k}\{h\}, \\ \infty & \text{otherwise.} \end{cases}$$

Note that, setting $\bar{\tau} = \infty$ will make the test before the Goldstein line search in Algorithm MOIF$_{front}$ surely satisfied so that the Goldstein line search is not executed. Furthermore, it is worth noting that, when $c_p = 1$, then the Goldstein line search is performed just as in the original version of MOIF$_{front}$, while if $c_p = 0$ then, as in DMS, no line search is ever executed.

The idea underlying the use of the above condition is that of managing the uniformity of the generated front by controlling the uniformity of the sampling step sizes related to the points of the list. As we can see from the performance profiles reported in figures 4.2 and 4.3, when we reduce c_p our algorithm became closer to DMS solver.

Finally, we observe that, when $c_p = 0$, Algorithm MOIF$_{front}$ becomes equal to a specific instance of DMS, namely the one using the set of directions $\{e_1, \ldots, e_n, -e_1, \ldots, -e_n\}$.

4.5.3 Comparison using a set of initial points

In this subsection, we compare MOIF$_{front}$ and DMS choosing a set of initial solutions rather than a single point. In particular, rather than starting the solvers from the single point x_0 as we did in subsections 4.5.1 and 4.5.2, we let them start from a list of n points equally spaced on the line segment

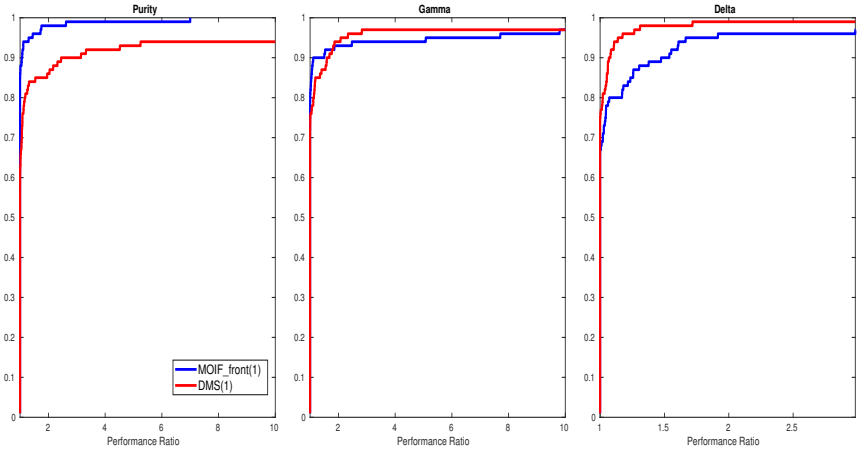

Figure 4.2: Comparison between $\texttt{MOIF}_{front}$, with $c_p = 0.5$, and DMS

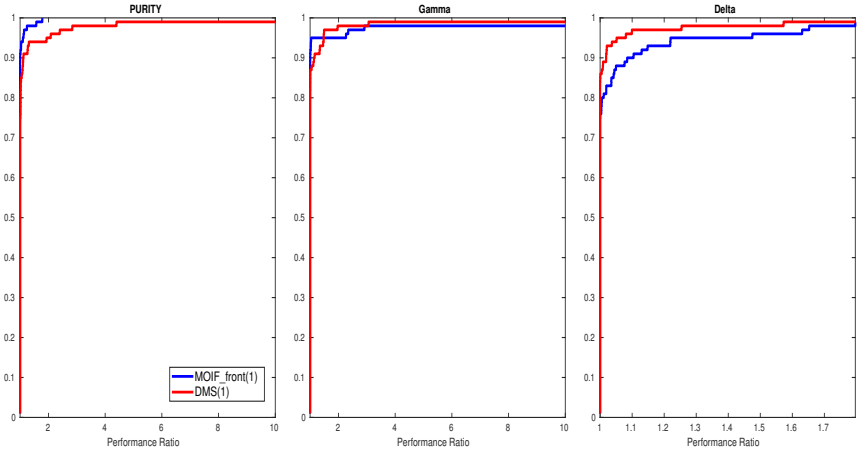

Figure 4.3: Comparison between $\texttt{MOIF}_{front}$, with $c_p = 0.2$, and DMS

joining the upper and lower bounds on the variables. Specifically, we let

$$\tilde{L}_0 = \{(x_0^1, h_0), \ldots, (x_0^n, h_0)\}$$

with

$$(x_0^j)_i = l_i + \frac{u_i - l_i}{n-1}(j-1), \quad \text{for} \quad i, j = 1, \ldots, n.$$

Then,

$$L_0 = \{(x_i, h_i) \in \tilde{L}_0 : \nexists\, (x_j, h_j) \in \tilde{L}_0 \text{ s.t. } F(x_j) \leq F(x_i)\},$$

i.e. the set obtained from $\tilde{L}_0$ by removing dominated solutions. Note that, for `DMS`, this is version `DMS`(n,line), as defined in [12].

The performance profiles are reported in figure 4.4. As in the previous case, we can still say that $\texttt{MOIF}_{front}$ outperforms `DMS` in terms of Purity. As for the Spread metrics, the two solvers are almost equivalent in terms of Spread Γ, while $\texttt{MOIF}_{front}$ is outperformed by `DMS` in terms of Spread Δ.

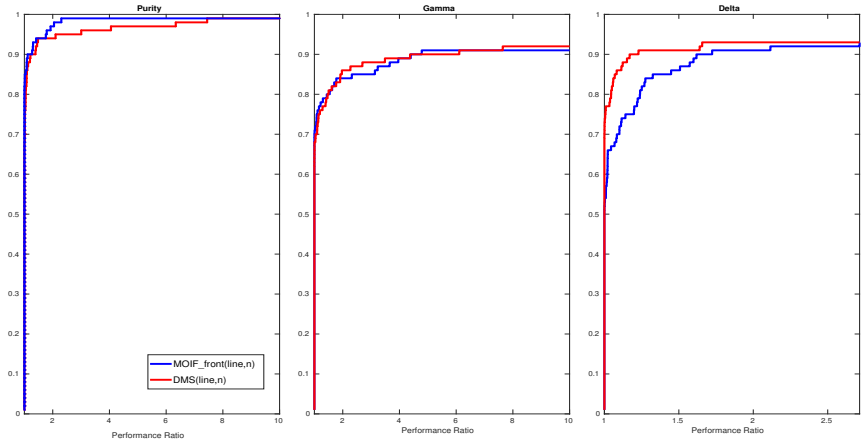

Figure 4.4: Comparison between $\texttt{MOIF}_{front}$, with $c_p = 1$, and `DMS` when both solvers start from set L_0

However, it is worth noting that the superiority of $\texttt{MOIF}_{front}$ over `DMS` in terms of Purity is considerably reduced with respect to the single starting point case (see figure 4.1).

This situation brings us to argue that the use of the initial set L_0 seems to help `DMS` to generate better estimates of the Pareto front. This is indeed

the case with DMS starting from set L_0 beating DMS starting from the centroid in terms of Purity but at the expense of the Spread metrics, for which the version of DMS starting from the centroid seems the best one. On the contrary, $\texttt{MOIF}_{front}$ seems not to be able to produce better estimates when it starts from the set L_0 rather than from the single point x_0. This can in turn be explained by recalling the 20,000 function evaluations limit. Indeed, at least for problems where the Implicit Filtering stepsize h_k is bigger than the tolerance 10^{-3} when the above limit is hit, the final estimate of the Pareto front could still be far away from the "real" one. This is to say that the line search techniques employed by MOIF have the effect to consume more function evaluations with respect to the "simpler" pattern search strategy of DMS.

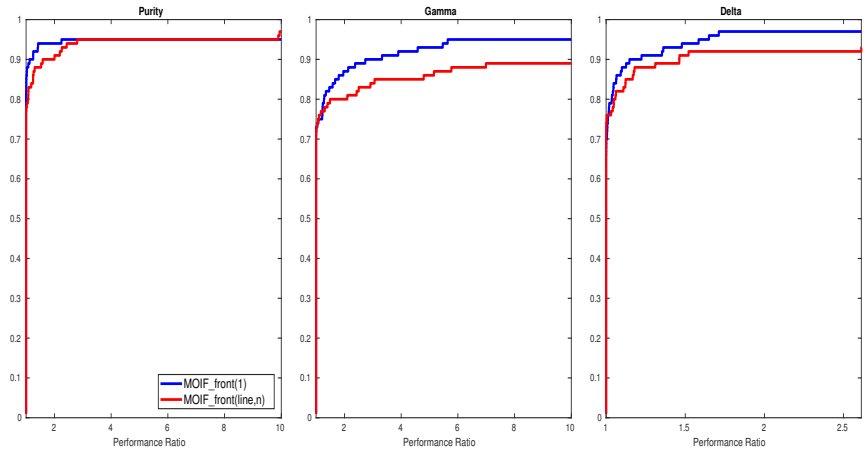

Figure 4.5: Comparison between $\texttt{MOIF}_{front}$ with $c_p = 1$, starting from the centroid of $\mathcal{F}$ and from set L_0

Finally, we have considered the comparison between the best version of $\texttt{MOIF}_{front}$, i.e. the one with $c_p = 1$ and starting from the centroid of the feasible region (as evidenced in Figure 4.5) and the default version of DMS, i.e. the one starting from set L_0. Performance profiles for the Purity and Spread metrics relative to this further comparison are reported in Figure 4.6.

From the figure, it is apparent the superiority of $\texttt{MOIF}_{front}$ (starting from the centroid) over DMS (starting from set L_0) both in terms of Purity and Spread Γ. As concerns the Spread Δ, it emerges quite a new situation with

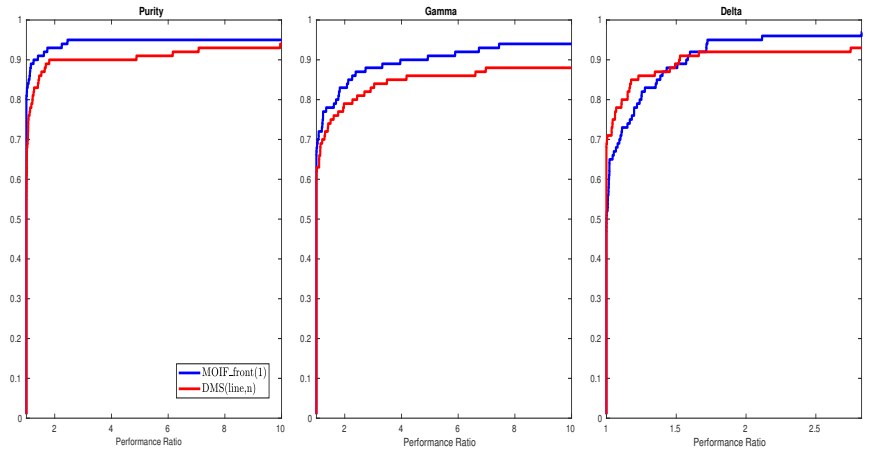

Figure 4.6: Comparison between $\texttt{MOIF}_{front}$, with $c_p = 1$ and starting from the centroid of $\mathcal{F}$, and DMS starting from set L_0

respect to the already seen comparisons. In fact, Spread Δ profiles reported in Figure 4.6 show that DMS is clearly less robust than $\texttt{MOIF}_{front}$ even though the former method is more efficient than the latter one.

4.5.4 Numerical results with noisy functions

In this section we report the numerical results obtained by $\texttt{MOIF}_{front}$ in the case of noisy functions.

In our experiments we consider additive noise sampled from a normal distribution. For every feasible point x, we evaluate a noisy version $\tilde{F} : \mathbb{R}^n \to \mathbb{R}^m$ of F such that

$$\tilde{F}(x) := F(x) + \mathcal{N}(0, \varepsilon\bar{\sigma}^2), \tag{4.25}$$

where $\varepsilon \in \mathbb{R}^+$ is a smoothing parameter which controls the noise level, $\bar{\sigma}^2 \in \mathbb{R}^m$ is such that

$$\bar{\sigma}^2 := \begin{bmatrix} |f_1^{max}| \\ \vdots \\ |f_m^{max}| \end{bmatrix}, \tag{4.26}$$

and, for each problem p, f_i^{max} is the maximum value of the $i-$th objective in the reference Pareto front F_p^{true} defined in (1.23).

In order to give a good estimation of the gap between the Pareto front found in the noisy case and the reference Pareto front, we consider the *Generational Distance* (GD) metric introduced in [48]:

$$GD = \frac{\left(\sum_{i=1}^{P} d_i^2\right)^{1/2}}{P}, \tag{4.27}$$

where P is the number of points found by a solver, $x_1, x_2, \ldots, x_P$, and d_i is the euclidean distance of the corresponding noise-free value $F(x_i)$ from the *real* Pareto front. GD is a recommended metric in test problems for which a set of Pareto optimal solutions is known. Although in our test problems the real Pareto front is generally unknown, we use it in order to show the extent of convergence of $\texttt{MOIF}_{front}$ and DMS with respect to their noise-free versions. Therefore we adopted in (4.27) the following definition of d_i:

$$d_i = \begin{cases} 0 & \text{if } x_i \text{ is non-dominated by } F_p^{true} \\ \min\limits_{x \in F_p^{true}} \{||F(x) - F(x_i)||_2\} & \text{otherwise.} \end{cases} \tag{4.28}$$

We remark that the contribution of d_i to the GD metric is zero not only if x_i belongs to the reference Pareto front, but also if it is better (this may happen because F_p^{true} is an approximation of the real Pareto Front). We also remark that GD, as the Purity metric, does not consider the quantity of generated points, but only the quality.

We tested both $\texttt{MOIF}_{front}$ and DMS, starting from the centroid of the feasible region $\mathcal{F}$ using the stopping criteria defined in section 4.5.2 and $\varepsilon \in \{0.0001, 0.05, 0.1, 0.2\}$. For each value of ε, 10 independent runs were executed for each test problem.

We did not observe significant differences in the performance of the two algorithms for $\varepsilon \in \{0.05, 0.1, 0.2\}$. Therefore, we report the results only for $\varepsilon = 0.0001$. Figure 4.7 shows the good performance of $\texttt{MOIF}_{front}$ in terms of GD metric compared with those of DMS. For the same value of $\varepsilon = 0.0001$ we have also compared $\texttt{MOIF}_{front}$ and DMS in terms of Purity and Spread metrics over the 1000 instances. The results of the comparison are reported in figure 4.8. We may observe that the results of the comparison are similar to those obtained without noise and reported in figure 4.1, that is, $\texttt{MOIF}_{front}$ outperforms DMS in terms of the Purity metric, while DMS outperforms $\texttt{MOIF}_{front}$ in terms of Spread metrics. On the whole, $\texttt{MOIF}_{front}$ and DMS show a similar robustness in presence of moderate noise.

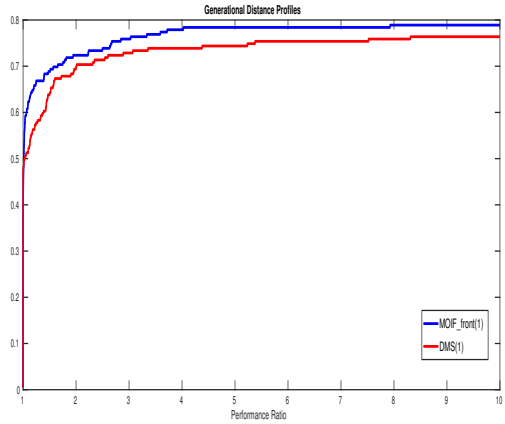

Figure 4.7: Generational Distance comparison between $\texttt{MOIF}_{front}$ and DMS, starting from the centroid, with $\varepsilon = 0.0001$

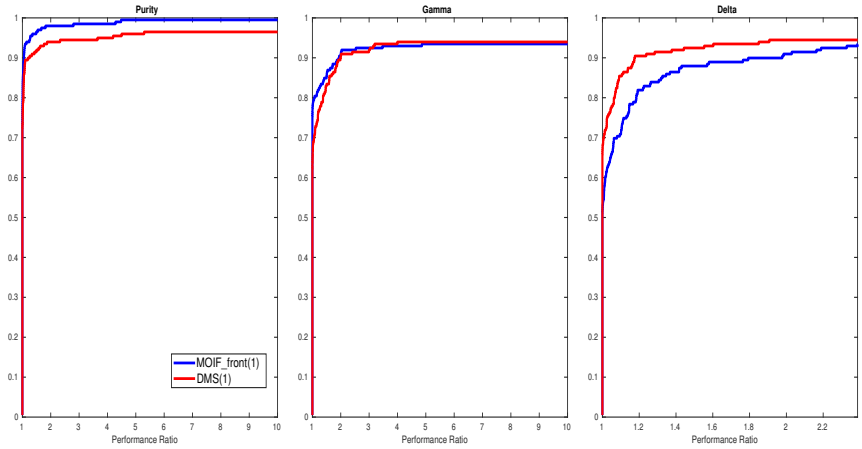

Figure 4.8: Purity and Spread metrics comparison between $\texttt{MOIF}_{front}$ and DMS, starting from the centroid, with $\varepsilon = 0.0001$

Chapter 5

Concluding remarks

Let us conclude our work, analyzing, for each considered topic, some possible future aspects to be evaluated. In Chapter 2, an *a posteriori* algorithm, based on the steepest descent framework, was proposed. Particularly, the steepest descent direction, computed by solving problem (2.2), is used for improving the current list with new non-dominated solutions. At this regard, two different Armijo-type line search techniques (see Algorithms 3 4) proven to generate new non-dominated solutions, are proposed.

Under some assumptions, global convergence properties to Pareto-stationary points of the algorithm were stated.

The proposed framework was compared with a multistart adaptation of the classical multiobjective steepest descent algorithm. Numerical results, obtained on a dataset of unconstrained multiobjective problems, showed the effectiveness of the proposed framework.

From a theoretical point of view, two possible generalizations of problem (2.2) may be defined. Firstly, since we are interested in enriching the list of new non-dominated solutions, one can iteratively compute the steepest descent direction using subsets of objectives.

Moreover, a further generalization can be done by abstracting the quadratic term in (2.2). At this regard, Multiobjective Newton and Quasi-Newton methods may be derived from this generalization. Sufficient descent conditions may be stated in order to define a *without preferences* globally convergent framework in terms of Pareto-stationarity.

In Chapter 3, problem (SMOP), which considers the ℓ_0-norm as an objective function, was taken into account. Continuously differentiable concave

functions were used to approximate the ℓ_0-norm. Equivalence properties, in terms of Pareto points, between the original problem and its concave approximation were stated.

Then, an *a posteriori* algorithm, namely `MOSO`, based on the steepest descent framework, was proposed. Numerical results, obtained on a dataset composed of portfolio selection problems, showed that `MOSO` has acceptable performance if compared to state-of-art multiobjective methods.

Problems like (SMOP) could be also used for modelling a Machine Learning *feature selection* problem in which the empirical risk and the number of active features have to be minimized. Datasets composed of feature selection problems could be used in order to validate the effectiveness of our algorithm.

Chapter 4 aimed to define a globally convergent extension of the Implicit Filtering algorithm for derivative-free MOO, namely `MOIF`. Under some assumptions on the objective functions, convergence properties to Pareto-stationary points were stated.

We tested `MOIF` on a dataset composed of 100 multiobjective problems with box constraints. Thanks to the line search phase, for all the considered problems `MOIF` generates always non-dominated points if compared to the ones found by state-of-art algorithms for derivative-free MOO.

Then, an *a posteriori* version, namely $\texttt{MOIF}_{front}$, is proposed. We tested $\texttt{MOIF}_{front}$ with on the same dataset using Purity and Spread metrics which are widely used in Pareto front evaluations. Noisy problems were considered in the analysis too.

Two different starting conditions were also tested. Very good results were obtained for what concerns the three considered metrics.

For what concerns future developments, the definition of *linked sequences* could be used in order to define global convergent properties of $\texttt{MOIF}_{front}$.

A further generalization of problem (1.4) could be defined adding some other general constraints to be treated as soft constraints i.e. through objective penalty functions.

Chapter 6

Appendix

6.1 Appendix: technical results

In the appendix we prove two technical results that are used for the convergence analysis.

Proposition 9. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be continuously differentiable and let $x \in \mathcal{F}$. Let $\{z_k\} \subset \mathcal{F}$ and $\{h_k\} \subset \mathbb{R}^+$ be sequences such that*

$$\lim_{k\to\infty} z_k = x \qquad \lim_{k\to\infty} h_k = 0. \tag{6.1}$$

Assume that, for $i = 1, \ldots, n$, at least one of the following condition holds

$$z_k + h_k e_i \in \mathcal{F},$$

$$z_k - h_k e_i \in \mathcal{F}.$$

Then we have

$$\lim_{k\to\infty} \nabla_{h_k} f(z_k) = \nabla f(x).$$

Proof. Let $i \in \{1, \ldots, n\}$ and define the following subsets

$$\begin{aligned} K_1 &= \{k : z_k + h_k e_i \in \mathcal{F},\ z_k - h_k e_i \notin \mathcal{F}\}, \\ K_2 &= \{k : z_k \pm h_k e_i \in \mathcal{F}\}, \\ K_3 &= \{k : z_k - h_k e_i \in \mathcal{F},\ z_k + h_k e_i \notin \mathcal{F}\}. \end{aligned}$$

By definition of approximated gradient we have

$$\frac{\partial_h f(z_k)}{\partial x_i} = \begin{cases} \frac{f(z_k+h_k e_i)-f(z_k)}{h_k} & k \in K_1 \\ \frac{f(z_k+h_k e_i)-f(z_k-h_k e_i)}{2h_k} & k \in K_2 \\ \frac{f(z_k)-f(z_k-h_k e_i)}{h_k} & k \in K_3 \end{cases}$$

Suppose that K_1 is an infinite subset. For all $k \in K_1$, by the Mean Value Theorem, we can write

$$\frac{\partial_h f(z_k)}{\partial x_i} = \frac{\partial f(\xi_k)}{\partial x_i},$$

where $\xi_k = z_k + \theta_k h_k e_i$, with $\theta_k \in (0,1)$. Taking the limits for $k \in K_1$ and $k \to \infty$, recalling (6.1) and the continuity of the gradient, we obtain

$$\lim_{k \in K_1, k \to \infty} \frac{\partial_h f(z_k)}{\partial x_i} = \frac{\partial f(x)}{\partial x_i}.$$

By repeating the same reasonings using the sets K_2 and K_3, we have

$$\lim_{k \to \infty} \frac{\partial_h f(z_k)}{\partial x_i} = \frac{\partial f(x)}{\partial x_i},$$

and the thesis is proved. □

Proposition 10. *Consider Problem* (4.7)*, let $F : \mathbb{R}^n \to \mathbb{R}^m$ be continuously differentiable, $x \in \mathcal{F}$, and let $\theta : \mathcal{F} \times R^+ \to \mathbb{R}$ be defined as in* (4.10)*. Then:*

(i) *$\theta(x,h) \leq 0$ for all $x \in \mathcal{F}$ and $h > 0$;*

(ii) *let $\{z_k\} \subset \mathcal{F}$ and $\{h_k\} \subset \mathbb{R}^+$ be sequences satisfying the assumptions of Proposition* 9*; we have*

$$\lim_{k \to \infty} \theta(z_k, h_k) = \theta(x).$$

Proof. (i) Given $x, y \in \mathcal{F}$ and $h > 0$, we consider the function g defined as follows:

$$g(y,h,x) = \max_{i=1,\ldots,m} \nabla_h f_i(x)^\top (y - x),$$

and note that

$$\theta(x,h) = \min_{y \in \mathcal{F}} g(y,h,x).$$

Then $\theta(x,h) \leq 0$ follows easily from $g(x,h,x) = 0$.

(ii) We preliminary observe that

$$|\max_i a_i - \max_i b_i| \leq \|a - b\|, \quad \text{for any } a, b \in \mathbb{R}^m. \tag{6.2}$$

Let us define

$$y(x) \in \arg\min_{y \in \mathcal{F}} \max_{i=1,\dots,m} \nabla f_i(x)^\top (y - x),$$

$$y_k \in \arg\min_{y \in \mathcal{F}} \max_{i=1,\dots,m} \nabla_{h_k} f_i(z_k)^\top (y - z_k),$$

so that

$$\max_{i=1,\dots,m} \nabla f_i(x)^\top (y(x) - x) \leq \max_{i=1,\dots,m} \nabla f_i(x)^\top (y_k - x)$$

$$\max_{i=1,\dots,m} \nabla_{h_k} f_i(z_k)^\top (y_k - z_k) \leq \max_{i=1,\dots,m} \nabla_{h_k} f_i(z_k)^\top (y(x) - z_k).$$

Denote by $J_{h_k}(z_k)$ the approximated Jacobian

$$J_{h_k}(z_k) = [\nabla_{h_k} f_1(z_k), \dots, \nabla_{h_k} f_m(z_k)]^\top.$$

We can write

$$\begin{aligned}
\theta(z_k, h_k) - \theta(x) &= \max_i \nabla_{h_k} f_i(z_k)^\top (y_k - z_k) - \max_i \nabla f_i(x)^\top (y(x) - x) \\
&\leq \max_i \nabla_{h_k} f_i(z_k)^\top (y(x) - z_k) - \max_i \nabla f_i(x)^\top (y(x) - x) \\
&\leq \|J_{h_k}(z_k)^\top (y(x) - z_k) - J(x)^\top (y(x) - x)\| \\
&\leq \|(J_{h_k}(z_k) - J(x))^\top y(x)\| + \\
&\quad + \|J(x)^\top x - J_{h_k}(z_k)^\top z_k + J(x)^\top z_k - J(x)^\top z_k\| \\
&\leq \|(J_{h_k}(z_k) - J(x))^\top y(x)\| + \\
&\quad + \|J(x)^\top (z_k - x)\| + \|(J_{h_k}(z_k) - J(x))^\top z_k\|.
\end{aligned}$$

A quite similar bound, with y_k in place of $y(x)$, can be obtained for $\theta(x) - \theta(z_k, h_k)$. Then, as z_k and y_k belong to the compact set $\mathcal{F}$, by Proposition 9, $|\theta(z_k, h_k) - \theta(x)| \to 0$ for $k \to \infty$. □

Chapter 7

Publications

This research activity has led to one publication in an international journals and three communications in conferences. These are summarized below.

International Journals

1. 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. Availale Online https://doi.org/10.1007/s10589-017-9953-2.

National Conferences

1. G. Cocchi, G. Liuzzi, A. Papini and M. Sciandrone, "An Implicit Filtering based algorithm for derivative free Multiobjective optimization", Società Italiana di Matematica Applicata e Industriale (SIMAI), Milan (Italy), September 13-16, 2016.
2. G. Cocchi, T. Levato, G. Liuzzi and M. Sciandrone, "A multiobjective approach for sparse mean-variance portfolios through a concave approximation", Optimization and Decision Science (ODS), Sorrento (Italy), September 4-7, 2017.
3. G. Cocchi, G. Liuzzi, S. Lucidi and M. Sciandrone, "On the convergence of steepest descent methods for multiobjective optimization", Optimization and Decision Science (ODS), Taormina (Italy), September 10-13, 2018.

Bibliography

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[2] F. Altiparmak, M. Gen, L. Lin, and T. Paksoy, "A genetic algorithm approach for multi-objective optimization of supply chain networks," *Comput. Ind. Eng.*, vol. 51, no. 1, pp. 196–215, Sep. 2006. [Online]. Available: http://dx.doi.org/10.1016/j.cie.2006.07.011

[3] E. Amaldi and V. Kann, "On the approximability of minimizing nonzero variables or unsatisfied relations in linear systems," *Theoretical Computer Science*, vol. 209, no. 1-2, pp. 237–260, 1998. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0304397597001151

[4] K. P. Anagnostopoulos and G. Mamanis, "A portfolio optimization model with three objectives and discrete variables," *Comput. Oper. Res.*, vol. 37, no. 7, pp. 1285–1297, Jul. 2010. [Online]. Available: http://dx.doi.org/10.1016/j.cor.2009.09.009

[5] C. Audet and J. E. Dennis, Jr., "Mesh adaptive direct search algorithms for constrained optimization," *SIAM J. on Optimization*, vol. 17, no. 1, pp. 188–217, Jan. 2006. [Online]. Available: http://dx.doi.org/10.1137/040603371

[6] C. Audet, G. Savard, and W. Zghal, "Multiobjective optimization through a series of single-objective formulations," *SIAM Journal on Optimization*, vol. 19, pp. 188–210, 2008. [Online]. Available: https://doi.org/10.1137/060677513

[7] ——, "A mesh adaptive direct search algorithm for multiobjective optimization," *European Journal of Operational Research*, vol. 204, no. 3, pp. 545 – 556, 2010. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0377221709008601

[8] K. Bailey and B. Fitzpatrick, "Estimation of groundwater flow parameters using least squares," *Mathematical and Computer Modelling*, vol. 26, no. 11, pp. 117–127, 1997. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0895717797002240

[9] S. Bandyopadhyay, S. K. Pal, and B. Aruna, "Multiobjective gas, quantitative indices, and pattern classification," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 34, no. 5, pp. 2088–2099, Oct 2004. [Online]. Available: https://doi.org/10.1109/TSMCB.2004.834438

[10] R. Brito and L. Vicente, "Efficient cardinality/mean-variance portfolios," in *System Modeling and Optimization*, ser. Springer series IFIP Advances in Information and Communication Technology, C. Pötzsche, C. Heuberger, B. Kaltenbacher, and F. Rendl, Eds. Berlin, Heidelberg: Springer, 2014, pp. 52–73.

[11] R. Carter, J. Gablonsky, A. Patrick, C. Kelley, and O. Eslinger, "Algorithms for noisy problems in gas transmission pipeline optimization," *Optimization and Engineering*, vol. 2, no. 2, pp. 139–157, Jun 2001. [Online]. Available: https://doi.org/10.1023/A:1013123110266

[12] A. Custódio, J. Madeira, A. Vaz, and L. Vicente, "Direct multisearch for multiobjective optimization," *SIAM Journal on Optimization*, vol. 21, no. 3, pp. 1109–1140, 2011. [Online]. Available: https://doi.org/10.1137/10079731X

[13] J. David, R. L. Ives, H. T. Tran, T. Bui, and M. E. Read, "Computer optimized design of electron guns," *IEEE Transactions on Plasma Science*, vol. 36, no. 1, pp. 156–168, Feb 2008. [Online]. Available: http://www.ncsu.edu/crsc/reports/ftp/pdf/crsc-tr07-13.pdf

[14] 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, Apr. 2002. [Online]. Available: http://dx.doi.org/10.1109/4235.996017

[15] K. Deb, M. Mohan, and S. Mishra, *Towards a Quick Computation of Well-Spread Pareto-Optimal Solutions*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2003.

[16] K. Doerner, W. J. Gutjahr, R. F. Hartl, C. Strauss, and C. Stummer, "Pareto ant colony optimization: A metaheuristic approach to multiobjective portfolio selection," *Annals of Operations Research*, vol. 131, no. 1, pp. 79–99, Oct 2004. [Online]. Available: https://doi.org/10.1023/B:ANOR.0000039513.99038.c6

[17] E. Dolan and J. Moré, "Benchmarking optimization software with performance profiles," *Mathematical programming*, vol. 91, no. 2, pp. 201–213, 2002. [Online]. Available: https://doi.org/10.1007/s101070100263

[18] L. Dos Santos Coelho and V. Mariani, "Combining of differential evolution and implicit filtering algorithm applied to electromagnetic design optimization," in *Soft Computing in Industrial Applications*. Springer, 2007, pp. 233–240.

[19] L. M. 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, pp. 201–216, Jan 2008. [Online]. Available: https://doi.org/10.1007/s10107-006-0071-7

[20] L. M. G. Drummond and B. F. Svaiter, "A steepest descent method for vector optimization," *Journal of Computational and Applied Mathematics*, vol. 175, no. 2, pp. 395 – 414, 2005. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0377042704003127

[21] J. Fliege, L. M. G. Drummond, and B. F. Svaiter, "Newton's method for multiobjective optimization," *SIAM Journal on Optimization*, vol. 20, no. 2, pp. 602–626, May 2009. [Online]. Available: http://dx.doi.org/10.1137/08071692X

[22] 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. [Online]. Available: https://doi.org/10.1007/s001860000043

[23] 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. [Online]. Available: https://doi.org/10.1137/15M1016424

[24] K. Fowler, C. Kelley, C. Kees, and C. Miller, "A hydraulic capture application for optimal remediation design," *Developments in Water Science*, vol. 55, pp. 1149–1157, 2004. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S016756480480131X

[25] K. Fowler, C. Kelley, C. Miller, C. Kees, R. Darwin, J. Reese, M. Farthing, and M. Reed, "Solution of a well-field design problem with implicit filtering," *Optimization and Engineering*, vol. 5, no. 2, pp. 207–234, 2004. [Online]. Available: https://doi.org/10.1023/B:OPTE.0000033375.33183.e7

[26] Y. Fu and U. M. Diwekar, "An efficient sampling approach to multiobjective optimization," *Annals of Operations Research*, vol. 132, no. 1, pp. 109–134, Nov 2004. [Online]. Available: https://doi.org/10.1023/B:ANOR.0000045279.46948.dd

[27] E. Fukuda and L. Drummond, "A survey on multiobjective descent methods," *Pesquisa Operacional*, vol. 34, no. 3, pp. 585–620, 2014. [Online]. Available: http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0101-74382014000300585&nrm=iso

[28] M. Gen, R. Cheng, and L. Lin, *Multiobjective Genetic Algorithms*. Springer, 2008, pp. 1–47. [Online]. Available: https://doi.org/10.1007/978-1-84800-181-7_1

[29] P. Gilmore and C. Kelley, "An implicit filtering algorithm for optimization of functions with many local minima," *SIAM Journal on Optimization*, vol. 5, no. 2, pp. 269–285, 1995.

[30] P. Gilmore, C. Kelley, C. Miller, and G. Williams, "Implicit filtering and optimal design problems," in *Optimal Design and Control*. Springer, 1995, pp. 159–176.

[31] M. Gong, J. Liu, H. Li, Q. Cai, and L. Su, "A multiobjective sparse feature learning model for deep neural networks," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 26, no. 12, pp. 3263–3277, 2015.

[32] Gurobi Optimization, LLC, "Gurobi optimizer reference manual," 2018. [Online]. Available: http://www.gurobi.com

[33] J. Jahn, "Scalarization in vector optimization," *Mathematical Programming*, vol. 29, no. 2, pp. 203–218, Jun 1984. [Online]. Available: https://doi.org/10.1007/BF02592221

[34] N. Jozefowiez, F. Semet, and E.-G. Talbi, "Multi-objective vehicle routing problems," *European Journal of Operational Research*, vol. 189, pp. 293–309, Jan. 2008. [Online]. Available: https://hal.archives-ouvertes.fr/hal-00831915

[35] A. Jüschke, J. Jahn, and A. Kirsch, "A bicriterial optimization problem of antenna design," *Computational Optimization and Applications*, vol. 7, no. 3, pp. 261–276, May 1997. [Online]. Available: https://doi.org/10.1023/A:1008611827855

[36] C. Kelley, *Implicit Filtering*. Society for Industrial and Applied Mathematics, 2011.

[37] L. Li, X. Yao, R. Stolkin, M. Gong, and S. He, "An evolutionary multiobjective approach to sparse reconstruction," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 6, pp. 827–845, 2014.

[38] C. J. Lin, S. Lucidi, L. Palagi, A. Risi, and M. Sciandrone, "Decomposition algorithm model for singly linearly constrained problems subject to lower and upper bounds," *Journal of Optimization Theory and Applications*, vol. 141, no. 1, pp. 107–126, 2009. [Online]. Available: https://doi.org/10.1007/s10957-008-9489-9

[39] 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. [Online]. Available: https://doi.org/10.1137/15M1037810

[40] S. Lucidi, L. Palagi, A. Risi, and M. Sciandrone, "A convergent decomposition algorithm for support vector machines," *Computational Optimization and Applications*, vol. 38, pp. 217–234, 2007. [Online]. Available: https://doi.org/10.1007/s10589-007-9044-x

[41] K. Miettinen, *Nonlinear Multiobjective Optimization*, ser. International Series in Operations Research & Management Science. Springer, 1998.

[42] ——, *Nonlinear multiobjective optimization*. Kluwer Academic Publishers, Boston, 1999.

[43] S. Obayashi, D. Sasaki, Y. Takeguchi, and N. Hirose, "Multiobjective evolutionary computation for supersonic wing-shape optimization," *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 2, pp. 182–187, July 2000.

[44] B. Ombuki, B. J. Ross, and F. Hanshar, "Multi-objective genetic algorithms for vehicle routing problem with time windows," *Applied Intelligence*, vol. 24, no. 1, pp. 17–30, Feb 2006. [Online]. Available: https://doi.org/10.1007/s10489-006-6926-z

[45] Z. Povalej, "Quasi-newton's method for multiobjective optimization," *J. Comput. Appl. Math.*, vol. 255, pp. 765–777, Jan. 2014. [Online]. Available: http://dx.doi.org/10.1016/j.cam.2013.06.045

[46] F. Rinaldi, F. Schoen, and M. Sciandrone, "Concave programming for minimizing the zero-norm over polyhedral sets," *Computational Optimization and Applications*, vol. 46, no. 3, pp. 467–486, 2010. [Online]. Available: https://doi.org/10.1007/s10589-008-9202-9

[47] K. C. Tan, Y. H. Chew, and L. H. Lee, "A hybrid multi-objective evolutionary algorithm for solving truck and trailer vehicle routing problems," *European Journal of Operational Research*, vol. 172, pp. 855–885, 2006. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0377221705000056

[48] D. A. Van Veldhuizen, "Multiobjective evolutionary algorithms: Classifications, analyses, and new innovations," Ph.D. dissertation, Wright Patterson AFB, OH, USA, 1999, aAI9928483.

[49] F. Wang, X. Lai, and N. Shi, "A multi-objective optimization for green supply chain network design," *Decision Support System*, vol. 51, no. 2, pp. 262–269, May 2011. [Online]. Available: http://dx.doi.org/10.1016/j.dss.2010.11.020

[50] X. Xue, K. W. E. Cheng, T. W. Ng, and N. C. Cheung, "Multi-objective optimization design of in-wheel switched reluctance motors in electric vehicles," *IEEE Transactions on Industrial Electronics*, vol. 57, pp. 2980–2987, 2010.

[51] W.-C. Yeh and M.-C. Chuang, "Using multi-objective genetic algorithm for partner selection in green supply chain problems," *Expert Syst. Appl.*, vol. 38, pp. 4244–4253, 04 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0957417410010481

[52] Q. Zhang, A. Zhou, S. Zhao, P. N. Suganthan, W. Liu, and S. Tiwari, "Multiobjective optimization test instances for the cec 2009 special session and competition (revised on 20/04/2009)," 2008.

[53] A. Zhou, B.-Y. Qu, H. Li, S.-Z. Zhao, P. Suganthan, and Q. Zhang, "Multiobjective evolutionary algorithms: A survey of the state of the art," *Swarm and Evolutionary Computation*, vol. 1, no. 1, pp. 32–49, 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2210650211000058