European Journal of Operational Research xxx (xxxx) xxx

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/eor

Continuous Optimization

New computable algorithms for smooth multiobjective optimization problems

Sorin-Mihai Grad [a,b,*], Tibor Illés [c], Petra Renáta Rigó [c]

[a] Faculty of Mathematics, University of Duisburg-Essen, Thea-Leymann-Str. 9, 45127 Essen, Germany
[b] Department of Mathematics, Babeș-Bolyai University Cluj-Napoca, Kogălniceanu Street No. 1, 400084 Cluj-Napoca, Romania
[c] Corvinus Centre for Operations Research at Corvinus Institute for Advanced Studies, Corvinus University of Budapest, Hungary

ARTICLE INFO

MSC:
90C29
49M05
58E17
65K05
90B50

Keywords:
Continuous optimization
Multiple objective programming
Joint decreasing direction
Linear programming
Weakly Pareto efficient solutions

ABSTRACT

We propose new practical algorithms for solving smooth multiobjective optimization problems based on determining joint decreasing directions via suitable linear programming problems. The presented iterative method is specialized for unconstrained, sign constrained and linearly constrained multiobjective optimization problems. In all cases discussed in this paper, we show that the objective function values sequence is decreasing with respect to the considered nonnegative orthant while the iterates are feasible. Furthermore, we prove that every accumulation point of the sequence generated by the algorithm, if any, is a substationary point to the considered multiobjective optimization problem, and, under convexity assumptions, it is actually a weakly Pareto efficient (also known as weakly Pareto-optimal) point. Different to similar algorithms from the literature, the ones proposed in this work involve joint decreasing directions that are easily computable in polynomial time by solving linear programming problems. Numerical experiments on unconstrained and linearly constrained convex multiobjective optimization test problems and on portfolio optimization problems show the applicability of the proposed algorithms.

1. Introduction

Multiobjective optimization problems consist in minimizing several objective functions at the same time, such problems arising in different fields, such as engineering, see Eschenauer et al. (1990), statistics, see Carrizosa and Frenk (1998), management science, see Evans (1984), etc. The most used solution notions for such problems are the Pareto and weakly Pareto efficient ones, see Boţ et al. (2009) and Luc (1989). A point is called Pareto efficient if there does not exist another point with the same or smaller objective function value (with respect to the corresponding nonnegative orthant in the image space of the objective function), such that there is a strict decrease in at least one objective function value. A point is called weakly Pareto efficient if there does not exist another point with strict decrease in all its objective values. These notions have also been extended to cases when other ordering cones are used instead of the nonnegative orthant and even to problems with infinitely-dimensional image space (Boţ et al., 2009).

One of the main approaches to handle multiobjective optimization problems is the scalarization technique, see Eichfelder (2008), Fliege and Svaiter (2000), Luc (1989) and Miettinen (1999). However, the scalar optimization problem attached to a multiobjective one often turns out to be unbounded (see, for instance, Bonnel et al. (2005, Remark 1) for a simple example), providing thus no valuable input in solving the original problem. In order to avoid fruitless trials, the multiobjective optimization community began exploring ways to produce algorithms capable of directly solving multiobjective optimization problems, most of which being direct extensions of iterative methods for tackling scalar optimization problems, such as the steepest descent method of Graña-Drummond and Svaiter (2005), the Barzilai–Borwein descent method of J. Chen et al. (2023), the conjugate gradient method of Lucambio Pérez and Prudente (2018), different versions of the projected gradient method by Bello Cruz et al. (2011), Fukuda and Graña-Drummond (2013, 2014), Graña-Drummond and Iusem (2004) and Mukai (1980), the conditional gradient method by Assunção et al. (2021) and W. Chen et al. (2023), Newton's method by Fliege et al. (2009), and even the proximal point algorithm, see Bonnel et al. (2005), Boţ and Grad (2018) and Grad (2019) in the nonsmooth case. Besides providing a single stationary/critical/(weakly) Pareto efficient point in each run, all these algorithms have as common feature the fact that there is a decrease (with respect to the partial ordering induced by the underlying cone) in the vector objective value at each iteration. Convergence results for the sequences produced by these methods were provided under different assumptions. While the determination

* Corresponding author at: Faculty of Mathematics, University of Duisburg-Essen, Thea-Leymann-Str. 9, 45127 Essen, Germany.
E-mail addresses: sorin-mihai.grad@uni-due.de (S.-M. Grad), tibor.illes@uni-corvinus.hu (T. Illés), petra.rigo@uni-corvinus.hu (P.R. Rigó).

https://doi.org/10.1016/j.ejor.2026.07.047
Received 13 June 2024; Accepted 28 July 2026
Available online 1 August 2026

of the descent direction plays a key role in these algorithms, in virtually all existing literature (except, maybe, for Assunção et al. (2021), where the feasible sets of the considered multiobjective optimization problems are taken compact) this step has a rather theoretical value, as the corresponding subproblems (that are the scalar optimization problems that need to be solved in each iteration of the considered iterative method) cannot be computationally solved without employing advanced numerical external solvers. Moreover, inexact versions of some of these methods were proposed in the literature in order to facilitate their implementation. Our aim is to propose new algorithms for solving different types of multiobjective optimization problems, where the descent directions are easily computable in polynomial time by means of linear programming problems. This fact supports our claim to deliver in this work practical and efficient algorithms for solving several classes of multiobjective optimization problems.

These new algorithms for solving multiobjective optimization problems are based on an idea from Illés and Lovics (2018). They proposed some techniques for determining joint decreasing directions for both unconstrained and linearly constrained multiobjective optimization problems. Afterward, a subdivision technique was employed for approximating the whole weakly Pareto efficient set of linearly constrained convex multiobjective optimization problems, and these theoretical achievements were illustrated by computational results obtained while solving the classical Markowitz model (Markowitz, 1952, 1959).

We propose a generic algorithm for solving multiobjective optimization problems where joint decreasing directions are used in order to guarantee descending values of the objective vector function. We discuss afterward the differences between it and other similar algorithms considered in the literature. In particular, we show that joint decreasing directions for these problems can be computed in polynomial time using results from linear programming, making thus our method practical in the sense that numerical results can be obtained easily for it. Then, we specialize the proposed algorithm for unconstrained, sign constrained and linearly constrained multiobjective optimization problems. In order to illustrate the applicability of the proposed methods, we provide numerical results. However, this paper is not an intensive numerical study, but a theoretical contribution that proposes an efficiently implementable approach for solving several classes of multiobjective optimization problems. From the test examples available in the literature we chose a convex unconstrained multiobjective optimization problem to highlight how our algorithm works. After that, we modified the selected test problem by adding linear constraints in such a way, that some of the weakly Pareto efficient points became infeasible for the constrained problem. The computational results in the constrained case show the effect of the linear constraints both on the running time and on the obtained weakly Pareto efficient set. Although this phenomenon is, intuitively, rather obvious, the influence of the additional constraints on the iterations of corresponding algorithm has not been illustrated and tested yet, up to our best knowledge.

Another feature that stresses the difference between the iterative methods we propose and their counterparts in the literature relies in the outcome of the subproblems they use for determining descent directions. While the latter require determining the unique optimal solution of a minimax quadratic optimization problem, the linear programming subproblems employed in the algorithms we introduce are dual degenerate, having thus infinitely many optimal solutions that deliver associated joint decreasing directions, of which it is enough to determine one in order to continue iterating. Moreover, in the case of constrained multiobjective optimization problems the previously known methods rely on projections on the feasible sets (that are rarely easy to perform from a computational point of view) or on solving constrained minimax quadratic optimization problems, while our approach ensures feasibility during the procedure of selecting joint decreasing directions without additional costs. Direct comparisons of the performance of the methods we propose with others from the literature for solving multiobjective optimization problems are beyond the scope of this work since, except for the conditional gradient method proposed in Assunção et al. (2021) (that is only able to solve multiobjective optimization problems with compact constraints), none of the theoretical algorithms introduced so far for solving multiobjective optimization problems seems to be fully implementable (for large-scale problems) because of numerical issues generated by the employment of advanced numerical solvers for dealing with their intermediate subproblems. Our work is not focused on optimizing the implementation of the proposed algorithms but on stressing their properties and flexibility, that provide space for more analyses on the numerical side, leaving performance improvement investigations for future studies.

The paper is organized as follows. In Section 2 some basic concepts and definitions related to multiobjective optimization problems are presented. In Section 3 we propose a generic scheme based on the computation of joint decreasing directions for solving multiobjective optimization problems, and we point out some differences between it and some of the similar algorithms from the literature. In Section 4 the new algorithm for unconstrained multiobjective optimization problems is defined. Section 5 is devoted to the algorithm for multiobjective optimization problems with sign constraints, while in Section 6 the new algorithm variant for linearly constrained multiobjective optimization problem is discussed. In all cases we show that the objective vector function values sequence is decreasing with respect to the natural partial ordering of $\mathbb{R}^m$. Furthermore, we prove that every accumulation point of the generated sequence, if any, is a substationary point, and, under convexity hypotheses imposed on the objective vector function and on the feasible set, a weakly Pareto efficient one. In Section 7 we provide numerical results for standard test multiobjective optimization problems in order to show the practical applicability of our new algorithms. Furthermore, using data from *Euro Stoxx 50* provided in Vörös et al. (2026), Vörös and Rappai (2025), we solved several portfolio optimization problems (see, for instance, Illés and Lovics (2018) and Markowitz (1952, 1959)) by implementing the proposed approach in the modeling language Mosel in FICO Xpress Optimizer (FICO, 2025). Section 8 contains concluding remarks and ideas for future research.

We use the following notations throughout the paper. Let $\mathbb{R}^m_{\oplus}$ and $\mathbb{R}^m_+$ be the nonnegative and the positive orthant in the m-dimensional Euclidean space ($m \geq 2$), respectively. We consider the partial ordering induced by $\mathbb{R}^m_+$, i.e. for $\mathbf{x}, \mathbf{y} \in \mathbb{R}^m$, $\mathbf{x} \leqq \mathbf{y}$ if $\mathbf{y} - \mathbf{x} \in \mathbb{R}^m_{\oplus}$ and $\mathbf{x} < \mathbf{y}$ if $\mathbf{y} - \mathbf{x} \in \mathbb{R}^m_+$. We write $\mathbf{x} \leq \mathbf{y}$ when $\mathbf{x} \leqq \mathbf{y}$ and $\mathbf{x} \neq \mathbf{y}$. We also use analogously for m-dimensional vectors the notations $\geqq$, $\geq$ and $>$. For real numbers we use the standard notations $<$, $>$, $\leq$, and $\geq$. Vectors are denoted by lowercase boldface Latin letters, $\mathbf{e}$ is the n-dimensional all-one vector. Furthermore, $\|\cdot\|$ denotes the standard Euclidean norm. Given two sets $A, B \in \mathbb{R}^m$, their Minkowski sum is denoted by $A+B$, and we write $A-B$ for $A+(-B)$. When $B = \{b\}$, we simplify the notation to $A \pm b$. Let $U \subseteq \mathbb{R}^n$ open and $F = (F_1, \ldots, F_m)^T : U \to \mathbb{R}^m$ continuously differentiable, i.e. $F \in C^1(U, \mathbb{R}^m)$. The gradient of F_j is denoted by ∇F_j ($j = 1, \ldots, m$), the Jacobian of F at $x \in U$ by $JF(x) \in \mathbb{R}^{m \times n}$ and the Hessian of F_j ($j = 1, \ldots, m$) at $x \in U$ is $\nabla^2 F_j(x)$. For simplicity, instead of calling F $\mathbb{R}^m_{\oplus}$-convex (on U) when F_j are all convex (on U) ($j = 1, \ldots, m$), we say that it is convex (on U).

2. Basic results in multiobjective optimization

In this section we introduce the general *multiobjective optimization problem* abbreviated MOP, for simplicity, from now on and the concepts of (weakly) Pareto efficient solutions. Let $\mathcal{F} \subseteq \mathbb{R}^n$ be closed with nonempty interior and $F = (F_1, \ldots, F_m)^T : \mathcal{F} \to \mathbb{R}^m$. The *general MOP* we consider is

$$\left.\begin{array}{l} \text{Min } F(\mathbf{x}) \\ \mathbf{x} \in \mathcal{F}. \end{array}\right\} \qquad (GMOP)$$

From now on we assume F to be continuously differentiable on $\operatorname{int}\mathcal{F}$. We call $(GMOP)$ *convex* when F and $\mathcal{F}$ are convex, and we

refer to such a situation as the *convex case*. We say that $\bar{\mathbf{x}} \in \mathcal{F}$ is a *substationary point* to $(GMOP)$ (see, for instance, Miettinen (1999)) if there exists a $\mathbf{w} \in S_m := \{\mathbf{w} \in \mathbb{R}^m_{\oplus} : \mathbf{e}^T\mathbf{w} = 1\}$, which satisfies $\mathbf{w}^T JF(\bar{\mathbf{x}}) = \mathbf{0}$. This notion is defined by means of a necessary condition for the weak Pareto efficiency of a point to $(GMOP)$ and its origin can be traced back to Kuhn and Tucker (1951), arguably the first paper on multiobjective optimization.

We say that $\bar{\mathbf{x}} \in \mathcal{F}$ is a *stationary* (*critical*) *point to* $(GMOP)$ if $\mathcal{R}(JF(\bar{\mathbf{x}})) \cap (-\mathbb{R}^m_+) = \emptyset$, where $\mathcal{R}(\cdot)$ denotes the image set of the considered mapping (in this case, the Jacobian of F at $\bar{\mathbf{x}}$). Hence, $\bar{\mathbf{x}} \in \mathcal{F}$ is stationary to $(GMOP)$ if and only if for all $\mathbf{v} \in \mathbb{R}^n$ we have $JF(\bar{\mathbf{x}})\mathbf{v} \not< 0$, that is for every $\mathbf{v} \in \mathbb{R}^n$ there exists a $j = j(\mathbf{v})$ such that

$$(JF(\bar{\mathbf{x}})\mathbf{v})_j = \nabla F_j(\bar{\mathbf{x}})^T\mathbf{v} \geq 0,$$

and this implies

$$\max_{i=1,\ldots,m} \nabla F_j(\bar{\mathbf{x}})^T\mathbf{v} \geq 0 \qquad \forall \mathbf{v} \in \mathbb{R}^n.$$

One can note that every stationary point to $(GMOP)$ is substationary to it, too.

Definition 2.1. We say that $\mathbf{x}^* \in \mathcal{F}$ is a

- *weakly Pareto efficient solution* to $(GMOP)$ if no feasible solution $\mathbf{x} \in \mathcal{F}$ exists satisfying $F(\mathbf{x}) < F(\mathbf{x}^*)$;
- *Pareto efficient solution* to $(GMOP)$ if no feasible solution $\mathbf{x} \in \mathcal{F}$ exists satisfying $F(\mathbf{x}) \leq F(\mathbf{x}^*)$.

Pareto solutions to $(GMOP)$ are also weakly Pareto ones to it, while the opposite implication does not hold in general. For a discussion on how to guarantee the latter, see Bonnel et al. (2005, Section 4). Since 1952, several methods have been proposed for finding *one* of the (weakly) Pareto efficient solutions to $(GMOP)$. Markowitz considered the following weighted optimization problem

$$\left.\begin{array}{l}\min \mathbf{w}^T F(\mathbf{x}) \\ \mathbf{x} \in \mathcal{F}\end{array}\right\} \qquad (WOP(\mathbf{w})),$$

where $\mathbf{w} \in S_m$. The next statement shows that $(WOP(\mathbf{w}))$ can be used for determining weakly Pareto efficient solutions to $(GMOP)$, see Boţ et al. (2009), Eichfelder (2008), Luc (1989) and Miettinen (1999).

Theorem 2.1. *Let $(GMOP)$ and the corresponding $(WOP(\mathbf{w}))$ be given for a $\mathbf{w} \in S_m$. Assume that $\mathbf{x}^* \in \mathcal{F}$ is an optimal solution to $(WOP(\mathbf{w}))$, then $\mathbf{x}^*$ is a weakly Pareto efficient solution to $(GMOP)$.*

For the completeness of the discussion we mention the following theorem, which shows that in the convex case each Pareto efficient solution to $(GMOP)$ can be determined by solving $(WOP(\mathbf{w}))$ defined by means of a suitable weights vector $\mathbf{w} \in S_m \cap \mathbb{R}^m_+$, see Luc (1989) and Miettinen (1999).

Theorem 2.2. *Let $(GMOP)$ be a convex MOP, and assume that $\mathbf{x}^* \in \mathcal{F}$ is a Pareto efficient solution to $(GMOP)$. Then there is a weight vector $\mathbf{w} \in S_m \cap \mathbb{R}^m_+$ and corresponding $(WOP(\mathbf{w}))$ whose optimal solution is $\mathbf{x}^*$.*

Remark 1. A point $\bar{\mathbf{x}} \in \mathcal{F}$ is substationary to $(GMOP)$ if and only if there exists a $\mathbf{w} \in S_m$ such that $\mathbf{w}^T\nabla F(\bar{\mathbf{x}}) = \mathbf{0}$. This is a necessary condition for the local optimality of $\bar{\mathbf{x}}$ to $(WOP(\mathbf{w}))$ and becomes sufficient for its (global) optimality when F is convex (on $\mathcal{F}$) and $\mathcal{F}$ is convex. Hence, by Theorem 2.1, when $(GMOP)$ is convex, its (sub)stationary points are also weakly Pareto efficient.

Now we present the definition of joint decreasing directions which will be used in this paper. This definition is based on the ideas given in Illés and Lovics (2018). For earlier discussions on joint decreasing directions see Schäffler et al. (2002).

Definition 2.2. Given $(GMOP)$ and a feasible point $\mathbf{x} \in \mathcal{F}$, a vector $\mathbf{v} \in \mathbb{R}^n \setminus \{\mathbf{0}\}$ is said to be a

- *joint decreasing direction* of F at point $\mathbf{x}$ if there exists $h_1 > 0$, such that, for every $h \in (0, h_1]$ one has $F(\mathbf{x} + h\mathbf{v}) < F(\mathbf{x})$;
- *feasible joint decreasing direction* of F if it is a joint decreasing direction of F at $\mathbf{x}$ and there exists $h_2 > 0$, such that, for every $h \in (0, h_2]$, $\mathbf{x} + h\mathbf{v} \in \mathcal{F}$ holds as well.

Remark 2. A vector $\mathbf{v} \in \mathbb{R}^n$ is a joint decreasing direction of F if and only if $JF(\mathbf{x})\mathbf{v} < \mathbf{0}$.

Let $\mathcal{D}_F(\mathbf{x}) = \{\mathbf{v} \in \mathbb{R}^n : JF(\mathbf{x})\mathbf{v} < \mathbf{0}\}$ be the set of all joint decreasing directions of F at $\mathbf{x}$. A method for determining them relies on employing the quadratic programming problem

$$\min_{\mathbf{w} \in S_m} \mathbf{w}^T \left(JF(\mathbf{x})JF(\mathbf{x})^T\right)\mathbf{w} \qquad (QOP(\mathbf{x})).$$

Theorem 2.3 (*cf. Schäffler et al. (2002), See Illés and Lovics (2018) for a more direct proof*)**.** *Let $(GMOP)$, a point $\mathbf{x} \in \mathbb{R}^n$, and the associated $(QOP(\mathbf{x}))$ be given. Let $F : \mathbb{R}^n \to \mathbb{R}^m$ be a continuously differentiable (and convex) vector function, and $\mathbf{w}^* \in \mathbb{R}^m$ denote the optimal solution to $(QOP(\mathbf{x}))$. We define the vector $\mathbf{q} = JF(\mathbf{x})^T\mathbf{w}^* \in \mathbb{R}^n$. If $\mathbf{q} = \mathbf{0}$, then $\mathbf{x}$ is a substationary (weakly Pareto efficient) point to $(GMOP)$, otherwise $-\mathbf{q}$ is a joint decreasing direction of F at point $\mathbf{x}$.*

A similar outcome can be reached by introducing, for $\mathbf{x} \in \mathbb{R}^n$, the following linear programming problem

$$\left.\begin{array}{l}\max\limits_{(\mathbf{q},q_0)\in\mathbb{R}^n\times\mathbb{R}} q_0 \\ JF(\mathbf{x})\mathbf{q} + q_0\mathbf{e} \leqq \mathbf{0} \\ 0 \leq q_0 \leq 1\end{array}\right\} \qquad (LP(\mathbf{x})).$$

Theorem 2.4 (*cf. Illés and Lovics (2018)*)**.** *Let $(GMOP)$, a point $\mathbf{x} \in \mathbb{R}^n$ and the associated $(LP(\mathbf{x}))$ be given. Let $F : \mathbb{R}^n \to \mathbb{R}^m$ be a continuously differentiable (and convex) vector function. Then $(LP(\mathbf{x}))$ always has an optimal solution $(\mathbf{q}^*, q_0^*)$. There are two cases for the optimal value of $(LP(\mathbf{x}))$, either $q_0 = 0$ when $\mathbf{x}$ is a substationary (weakly Pareto efficient) point to $(GMOP)$, or $q_0 = 1$ in which case $\mathbf{q}^*$ is a joint decreasing direction of F at $\mathbf{x}$.*

Remark 3. One might ask what is the role of the constraint $0 \leq q_0 \leq 1$ in the linear programming problem $(LP(\mathbf{x}))$. As Theorem 2.4 reveals, when $q_0 = 0$, a substationary (or weakly Pareto efficient in the convex case) solution to $(GMOP)$ is found, while the situation $q_0 = 1$ delivers a joint decreasing direction of F at $\mathbf{x}$. The constraint $0 \leq q_0 \leq 1$ ensures the finiteness of the optimal solution to $(LP(\mathbf{x}))$. This, together with the feasibility of the considered linear programming subproblem and the fact that the latter is a maximization problem ensures its solvability. That is why the upper bound on q_0 is necessary.

3. Generic scheme for solving general multiobjective optimization problems

Based on the above discussion, we propose in Algorithm 3.1 a generic scheme for solving GMOPs that makes use of joint decreasing directions. When $\mathcal{F} = \mathbb{R}^n$, in Step 2 one needs to determine a joint decreasing direction.

Let us briefly go through the steps of Algorithm 3.1. After the initialization, the method checks whether there are (feasible) joint decreasing directions of F at the current iterate $\mathbf{x}^k$. In case there are not, the stopping criterion is activated, as $\mathbf{x}^k$ is a substationary solution to $(GMOP)$. Otherwise, a (feasible) joint decreasing direction of F is determined (based on Theorem 2.4 as we explain later), and $\mathbf{u}^k$ is computed. The next step provides an iterative procedure to determine the stepsize t_k (involving $\mathbf{u}^k$), while the final step gives the next iterate $\mathbf{x}^{k+1}$ and returns the algorithm to the verification step.

Algorithm 3.1 Generic scheme for solving GMOPs

[Step 1] take $\mathbf{x}^0 \in \mathcal{F} \subseteq \mathbb{R}^n$, $0 < \beta < 1$, $k = 0$, $J = \left\{ \frac{1}{2^n} : n \in \mathbb{N} \right\}$

[Step 2] **if** $\mathcal{D}_F(\mathbf{x}^k) = \emptyset$ **then** STOP ($\mathbf{x}^k$ is a substationary solution to $(GMOP)$);

otherwise select a *feasible joint decreasing direction* $\mathbf{v}^k \in \mathcal{D}_F(\mathbf{x}^k)$ and compute $\mathbf{u}^k = JF(\mathbf{x}^k)\,\mathbf{v}^k$

[Step 3] choose t_k as the largest $t \in J$ s.t. $F(\mathbf{x}^k + t\,\mathbf{v}^k) \leqq F(\mathbf{x}^k) + \beta t\,\mathbf{u}^k$

[Step 4] let $\mathbf{x}^{k+1} = \mathbf{x}^k + t_k\,\mathbf{v}^k$ and $k \leftarrow k+1$; go to Step 2

Both for reader's convenience and for illustrative purposes, we recall in the following subsections several full algorithms from the literature for solving GMOPs. They seem to be similar and the generic scheme for solving GMOPs Algorithm 3.1 presented above looks similar to them, too, however, the key difference between these methods can be found in the way the descent directions are computed (the second step of Algorithm 3.1 and its counterparts recalled in the following). Below, we point out in each case what is this step and how it differs from the one in Algorithm 3.1.

3.1. Algorithms for solving unconstrained multiobjective optimization problems

First, we present an overview on the algorithms for solving unconstrained MOPs proposed in the literature, see also Fukuda and Graña-Drummond (2014).

Steepest descent algorithm

We begin with the steepest descent algorithm introduced in Fliege and Svaiter (2000). It is proven that the method converges to a critical point to $(GMOP)$. In Graña-Drummond and Svaiter (2005), the algorithm is refined in order to solve smooth unconstrained MOPs, where the vector-minimization is considered with respect to a closed convex ordering cone. The authors proved that every accumulation point of the generated sequence is critical. The sketch of the steepest descent algorithm used in these papers is presented in Algorithm 3.2.

Algorithm 3.2 Steepest descent algorithm

[Step 1] take $\mathbf{x}^0 \in \mathcal{F} = \mathbb{R}^n$, $0 < \beta < 1$, $k = 0$, $J = \left\{ \frac{1}{2^n} : n \in \mathbb{N} \right\}$

[Step 2] determine the steepest descent direction $\mathbf{v}^k = \operatorname{argmin}_{\mathbf{v}\in\mathbb{R}^n} \left\{ \varphi_{\mathbf{x}^k}(\mathbf{v}) + \frac{1}{2}\|\mathbf{v}\|^2 \right\}$ and $\theta_k = \varphi_{\mathbf{x}^k}(\mathbf{v}^k) + \frac{1}{2}\|\mathbf{v}^k\|^2$, where $\varphi_x(\mathbf{v}) = \max\limits_{j=1,\dots,m} \nabla F_j(\mathbf{x})^T\mathbf{v}$; if $\theta_k = 0$: STOP

[Step 3] choose t_k as the largest $t \in J$ s.t. $F(\mathbf{x}^k + t\mathbf{v}^k) \leqq F(\mathbf{x}^k) + \beta t J F(\mathbf{x}^k)\mathbf{v}^k$

[Step 4] let $\mathbf{x}^{k+1} = \mathbf{x}^k + t_k\mathbf{v}^k$ and $k \leftarrow k+1$; go to Step 2

Step 2 of Algorithm 3.1 and its counterpart in Algorithm 3.2 are different, as the first one delivers a joint decreasing direction, while the other a steepest descent direction obtained by solving the minimax quadratic optimization problem $\min_{\mathbf{v}\in\mathbb{R}^n} \left\{ \varphi_{\mathbf{x}^k}(\mathbf{v}) + \frac{1}{2}\|\mathbf{v}\|^2 \right\}$. Moreover, Algorithm 3.2 requires the additional computation of θ_k, $k \geq 0$. Last but not least, the stopping criteria of the two algorithms differ as well.

Conjugate gradient algorithm

In Lucambio Pérez and Prudente (2018) one can find nonlinear conjugate gradient methods for finding critical points of vector functions with respect to a partial ordering induced by a closed convex pointed cone with nonempty interior. No convexity assumption is imposed on the objective vector function. Under inexact line search, the authors proved that the sequences generated by the method find critical points to $(GMOP)$. The sketch of their basic algorithm is given in Algorithm 3.3, particularized for the framework considered in our paper.

Comparing Algorithms 3.1 and 3.3, the key difference can be found in Step 2, where the descent direction is computed in each case in a specific way. Furthermore, Algorithm 3.3 requires an additional parameter sequence $(\sigma_k)_k > 0$, and the stopping criteria of the two iterative methods differ as well. As it can be seen, Step 2 in the algorithm from Lucambio Pérez and Prudente (2018) uses a complicated and sophisticated minmax problem to find the next joint decreasing direction. In the minimization part, the regularization term ensures the uniqueness of $\mathbf{v}^k$. However, from a computational point of view, such nonlinear optimization problem can be solved only up to ε-optimality, that may cause some practical issues.

Algorithm 3.3 Conjugate gradient algorithm

[Step 1] take $\mathbf{x}^0 \in \mathcal{F} = \mathbb{R}^n$, $0 < \beta < 1$, $\sigma_j > 0$, $j \in \mathbb{N}$, $k = 0$, $J = \left\{ \frac{1}{2^n} : n \in \mathbb{N} \right\}$

[Step 2] determine the descent direction $\mathbf{v}^k = \operatorname{argmin}_{\mathbf{v}\in\mathbb{R}^n} \left\{ \varphi_{\mathbf{x}^k}(\mathbf{v}) + \frac{1}{2}\|\mathbf{v}\|^2 \right\}$, where $\varphi_x(\mathbf{v}) = \max\limits_{j=1,\dots,m} \nabla F_j(\mathbf{x})^T\mathbf{v}$; if $\mathbf{v}^k = \mathbf{0}$: STOP; define $\mathbf{d}^k = \mathbf{d}^{k-1} + \sigma_k\mathbf{v}^k$ ($\mathbf{d}^0 = \mathbf{v}^0$)

[Step 3] choose t_k as the largest $t \in J$ s.t. $F(\mathbf{x}^k + t\mathbf{d}^k) \leqq F(\mathbf{x}^k) + \beta t J F(\mathbf{x}^k)\mathbf{d}^k$

[Step 4] let $\mathbf{x}^{k+1} = \mathbf{x}^k + t_k\mathbf{d}^k$ and $k \leftarrow k+1$; go to Step 2

Note that all the methods introduced for constrained optimization listed in the next subsection work for the unconstrained case, too.

3.2. Algorithms for solving constrained multiobjective optimization problems

Further, we recall the iterative methods proposed in the literature for solving constrained MOPs.

Newton's method

The extension of Newton's method proposed in Fliege et al. (2009) for solving GMOPs follows. The objective vector functions are supposed to be twice continuously differentiable and locally strongly convex, while the feasible set $\mathcal{F}$ is taken convex. Under these hypotheses the method is locally superlinearly convergent to Pareto efficient points when the objective vector function is cone-convex. Moreover, in Graña-Drummond et al. (2014) an improvement is proposed, where quadratic convergence of the generated sequence is achieved. Below we present the scheme of multiobjective Newton's method in Algorithm 3.4.

Algorithm 3.4 Newton's method

[Step 1] take $\mathbf{x}^0 \in \mathcal{F} \subseteq \mathbb{R}^n$, $0 < \beta < 1$, $k = 0$, $J = \left\{ \frac{1}{2^n} : n \in \mathbb{N} \right\}$

[Step 2] determine the Newton direction $\mathbf{s}^k = \operatorname{argmin}_{\mathbf{s}\in\mathbb{R}^n} \max\limits_{j=1,\dots,m} \left\{ \nabla F_j(\mathbf{x}^k)^T\mathbf{s} + \frac{1}{2}\mathbf{s}^T\nabla^2 F_j(\mathbf{x}^k)\mathbf{s} \right\}$

and $\theta_k = \inf\limits_{\mathbf{s}\in\mathbb{R}^n} \max\limits_{j=1,\dots,m} \left\{ \nabla F_j(\mathbf{x}^k)^T\mathbf{s} + \frac{1}{2}\mathbf{s}^T\nabla^2 F_j(\mathbf{x}^k)\mathbf{s} \right\}$; if $\theta_k = 0$: STOP

[Step 3] choose t_k as the largest $t \in J$ s.t. $\begin{cases} \mathbf{x}^k + t\mathbf{s}^k \in \mathcal{F} \\ F_j(\mathbf{x}^k + t\mathbf{s}^k) \leq F_j(\mathbf{x}^k) + \beta t\theta_k, \ j = 1, \dots, m \end{cases}$

[Step 4] let $\mathbf{x}^{k+1} = \mathbf{x}^k + t_k\mathbf{s}^k$ and $k \leftarrow k+1$; go to Step 2

Even though Newton's algorithm is a second order method, while the other ones considered in this paper are first order ones, its general scheme is consistent with the others. Of course, Step 2 in Algorithm 3.4 is not the one in Algorithm 3.1 since Newton's direction is obtained by minimizing the max-ordering scalarization of the variations on the quadratic approximation of the objective vector function. Furthermore, Step 3 and the stopping criteria differ in this case, too, and, as mentioned above, Newton's method requires stronger hypotheses imposed on the objective vector function.

Projected gradient method

A projected gradient algorithm for solving smooth MOPs consisting in vector-minimizing over a closed convex set $\mathcal{F} \subseteq \mathbb{R}^n$ a vector function $F : \mathcal{F} \to \mathbb{R}^m$ with respect to the partial ordering induced by a closed

convex cone was proposed in Graña-Drummond and Iusem (2004). They showed that any accumulation point of the sequence generated by this method converges to a stationary point to the considered problem. In the convex case and under some mild assumptions they also show convergence of the sequence generated by the algorithm to a weakly Pareto efficient solution to the considered problem. Improvements and extensions of the method were proposed in Fukuda and Graña-Drummond (2011, 2013). The sketch of the method adapted to the present framework is presented in Algorithm 3.5.

Algorithm 3.5 Projected gradient algorithm

[Step 1] take $\mathbf{x}^0 \in \mathcal{F} \subseteq \mathbb{R}^n$, $0 < \beta < 1$, $\sigma > 0$, $k = 0$, $J = \left\{ \frac{1}{2^n} : n \in \mathbb{N} \right\}$

[Step 2] determine the descent direction $\mathbf{v}^k = \operatorname{argmin}_{\mathbf{v} \in \mathcal{F} - \mathbf{x}^k} \left\{ \sigma \varphi_{\mathbf{x}^k}(\mathbf{v}) + \frac{1}{2}\|\mathbf{v}\|^2 \right\}$ and $\theta_k = \varphi_{\mathbf{x}^k}(\mathbf{v}^k) + \frac{1}{2}\|\mathbf{v}^k\|^2$, where $\varphi_{\mathbf{x}}(\mathbf{v}) = \max_{j=1,\ldots,m} \nabla F_j(\mathbf{x})^T \mathbf{v}$; if $\theta_k = 0$: STOP

[Step 3] choose t_k as the largest $t \in J$ s.t. $F(\mathbf{x}^k + t\mathbf{v}^k) \leqq F(\mathbf{x}^k) + \beta t J F(\mathbf{x}^k)\mathbf{v}^k$

[Step 4] let $\mathbf{x}^{k+1} = \mathbf{x}^k + t_k \mathbf{v}^k$ and $k \leftarrow k + 1$; go to Step 2

The differences between Algorithms 3.1 and 3.5 are basically the same like between the first one and Algorithm 3.2, with the additional difficulty that the steepest descent direction in Step 2 of Algorithm 3.5 is chosen so that the new iterate is feasible, and Algorithm 3.5 also uses a parameter σ.

Conditional gradient method

When the feasible set $\mathcal{F} \subseteq \mathbb{R}^n$ is taken to be convex and compact, several algorithms for solving ($GMOP$) that contain simpler intermediate subproblems (i.e. scalar optimization problems to be solved in Step 2) than the one considered in Step 2 of Algorithm 3.5 were proposed in Assunção et al. (2021), Mukai (1980) and W. Chen et al. (2023). We recall below as Algorithm 3.6 the conditional gradient (Frank–Wolfe) method introduced in Assunção et al. (2021) (and extended in W. Chen et al. (2023) for vector-minimization with respect to a closed convex ordering cone).

Algorithm 3.6 Conditional gradient algorithm

[Step 1] take $\mathbf{x}^0 \in \mathcal{F} \subseteq \mathbb{R}^n$, $0 < \beta < 1$, $k = 0$, $J = \left\{ \frac{1}{2^n} : n \in \mathbb{N} \right\}$

[Step 2] determine the descent direction $\mathbf{v}^k \in -\mathbf{x}^k + \operatorname{argmin}_{\mathbf{d} \in \mathcal{F}} \max_{j=1,\ldots,m} \nabla F_j(\mathbf{x}^k)^T(\mathbf{d} - \mathbf{x}^k)$;
if $\max_{j=1,\ldots,m} \nabla F_j(\mathbf{x}^k)^T \mathbf{v}^k = 0$: STOP

[Step 3] choose t_k as the largest $t \in J$ s.t. $F(\mathbf{x}^k + t\mathbf{v}^k) \leqq F(\mathbf{x}^k) + \beta t J F(\mathbf{x}^k)\mathbf{v}^k$

[Step 4] let $\mathbf{x}^{k+1} = \mathbf{x}^k + t_k \mathbf{v}^k$ and $k \leftarrow k + 1$; go to Step 2

The differences between Algorithms 3.1 and 3.6 seem to be more subtle than the ones between the first one and the previously recalled iterative methods, however, one can note at least two fundamental ones. While the compactness of the feasible set is crucial for ensuring the solvability of the subproblems employed in Algorithm 3.6, Algorithm 3.1 requires no such restrictive hypotheses. Moreover, in Assunção et al. (2021) it is indicated that in order to use a "linear optimization oracle" for solving these subproblems transformed into linear programming ones, the feasible sets of the latter need to be polyhedral compact, i.e. polytopes. The approach we propose does not require adding such additional hypotheses in order to determine the feasible joint decreasing directions.

Remark 4. The algorithms for solving MOPs recalled in this section deliver one critical or weakly Pareto efficient solution to the considered problem in each run. In general, most of the algorithms starting from one feasible solution find one (weakly) Pareto efficient solution. Regarding how to go from one such solution to the whole (weakly) Pareto frontier, one can identify in the literature mainly two views, leading to *inner and outer approximations* of (weakly) Pareto efficient sets.

The inner approximations are based on the idea that one starts some algorithm from multiple feasible solutions and ends up with multiple (weakly) Pareto efficient solutions. Up to our best knowledge, in these inner approximations, there is no control on what type of (weakly) Pareto efficient solutions would be found. A further weakness of this approach is that it leads to finitely many Pareto efficient solutions with extremely large computational effort.

The outer approximations are described in the paper of Illés and Lovics (2018), where the authors have also provided an iterative algorithm to find ε-approximations of the (weakly) Pareto efficient set of the considered MOPs. The advantage of the method is that such ε-approximations for smaller-sized, convex MOPs can be computed. The disadvantage is that this approach needs an unpredictably large number of starting feasible solutions during the approximation process. Therefore, good outer ε-approximations of (weakly) Pareto optimal efficient sets for small $\varepsilon > 0$ and large, convex MOPs currently are unlikely to be obtained in practice.

To sum up, the algorithms presented in this section show that the determination of the descent directions and the solutions of the subproblems appearing in Step 2 (in each case) plays a crucial role in these methods. However, out of the previously mentioned papers only in few cases one can read about numerical results or implementation techniques, and, in those cases, inexact versions of these methods were considered. Not even unconstrained MOPs can be computationally solved via these methods without resorting to advanced numerical external solvers, and the constrained case further complicates things (as additional restrictive hypotheses, such as compactness, or potentially difficult projections are required). In particular, guaranteeing the feasibility of the iterates is a cumbersome task in the constrained case, that can overcomplicate the subproblems, in particular when the current iterate lies on the boundary of the feasible set. As none of the previously proposed algorithms for solving constrained MOPs took into consideration the structure of the corresponding feasible sets, these iterative methods are not practical from a computational point of view. Our aim is to introduce new algorithms based on Algorithm 3.1 where the subproblems are easily solvable in an efficient and exact manner. In the following section we propose a new algorithm for solving unconstrained MOPs. After that, we present other variants of the new algorithm for solving smooth MOPs with sign constraints and linear constraints, respectively. Different to the mentioned methods, our approach does not involve additional efforts or costs when dealing with constraints when the (feasible) joint decreasing directions are determined. Consequently, the algorithms we propose are always fully implementable for computationally solving both unconstrained and constrained MOPs, even for large scale problems, by exploiting the well-developed apparatus of linear programming.

4. New algorithm for solving unconstrained multiobjective optimization problems

In this section we present the new algorithm we propose for solving unconstrained MOPs which involves linear programming subproblems. The main steps of the method are given in Algorithm 4.1. While Algorithm 3.1 is a generic algorithmic scheme, Algorithm 4.1 is a concrete iterative method for solving unconstrained MOPs.

Following the result from Illés and Lovics (2018) we examine the case (using notations from Step 2 of Algorithm 4.1)

$$JF(\mathbf{x})\mathbf{q} + q_0\mathbf{e} \leqq \mathbf{0}, \quad q_0 > 0. \tag{2}$$

If system (2) has a solution $(\mathbf{q}, q_0)$, then $\left((1/q_0)\mathbf{q}, 1\right)$ solves it, too, so the optimal value of the objective function in Step 2 of Algorithm 4.1 is 1, in which case $q_0 = 1$ in Step 3. This means that

$$JF(\mathbf{x})\mathbf{q} \leqq -\mathbf{e}$$

so $\mathbf{q}$ is a *joint decreasing direction* of the vector function F.

Algorithm 4.1 New algorithm for solving unconstrained MOPs

[Step 1] take $\mathbf{x}^0 \in \mathcal{F} = \mathbb{R}^n$, $0 < \beta < 1$, $k = 0$, $J = \left\{ \frac{1}{2^n} : n \in \mathbb{N} \right\}$

[Step 2] determine the *joint decreasing direction* $\mathbf{q} \in \mathbb{R}^n$ by solving

$$\begin{aligned} &\max_{(\mathbf{q},q_0)\in\mathbb{R}^n\times\mathbb{R}} q_0 \\ &JF(\mathbf{x}^k)\mathbf{q} + q_0\,\mathbf{e} \leqq \mathbf{0} \\ &0 \leq q_0 \leq 1 \end{aligned} \tag{1}$$

[Step 3] if $q_0 = 0$ then STOP ($\mathbf{x}^k$ is a substationary solution to $(GMOP)$); otherwise $\mathbf{v}^k := \mathbf{q}$;

[Step 4] choose t_k as the largest $t \in J$ s.t. $F(\mathbf{x}^k + t\,\mathbf{v}^k) \leqq F(\mathbf{x}^k) + \beta t\, JF(\mathbf{x}^k)\mathbf{v}^k$

[Step 5] let $\mathbf{x}^{k+1} = \mathbf{x}^k + t_k\,\mathbf{v}^k$ and $k \leftarrow k+1$; go to Step 2

Remark 5. In practice, one needs not determine (computationally) an optimal solution to the problem (1) as it is enough to reach a single iteration where $q_0 > \zeta$, with $\zeta \in (0.1, 1)$ and $\mathbf{q} \neq \mathbf{0}$ in order to decide that the stopping criterion of Algorithm 4.1 is not activated and the linear programming solver can be stopped, as (2) is solvable, and $\left((1/q_0)\mathbf{q}, 1\right)$ is a joint decreasing direction that can be used for constructing the next iterate.

This observation opens possibilities for accelerating the computations when implementing Algorithm 4.1, however, such investigations are beyond the scope of this paper, since our aim is to propose an efficiently implementable class of algorithms for solving MOPs. In Section 7 we actually show the validity of our claim. How can these algorithms be optimized is surely a relevant question that can be addressed in subsequent studies.

If system (2) has no solution, then the optimal value of the objective function of (1) is 0 (as $(\mathbf{0}, 0)$ is feasible to the considered linear programming problem), and from a variant of the *Farkas Lemma* (see Illés and Lovics (2018)) we know that there exists a $\mathbf{w} \in \mathbb{R}^m$ which satisfies the following linear inequality system

$$\mathbf{w}^T JF(\mathbf{x}) = \mathbf{0}, \quad \mathbf{e}^T\mathbf{w} = 1, \quad \mathbf{w} \geqq \mathbf{0}. \tag{3}$$

This means that if the optimal value of the problem $(LP(\mathbf{x}))$ is 0, then $\mathbf{x}$ is a *substationary point* to $(GMOP)$.

Lemma 4.1. *Let $(GMOP)$ be given with $\mathcal{F} = \mathbb{R}^n$ and the vector function $F : \mathbb{R}^n \to \mathbb{R}^m$ continuously differentiable. Assumming that Algorithm 4.1 starting at $\mathbf{x}^0 \in \mathcal{F}$ produces for $0 < \beta < 1$ the sequence of points $\{\mathbf{x}^k\} \subseteq \mathbb{R}^n$, then*

$$F(\mathbf{x}^{k+1}) < F(\mathbf{x}^k), \qquad \forall k \geq 0. \tag{4}$$

Proof. Let $k \geq 0$. Since $\mathbf{v}^k$ is a joint decreasing direction computed in Step 3 of Algorithm 4.1, we have $JF(\mathbf{x}^k)\,\mathbf{v}^k \leqq -\mathbf{e}$. Therefore, in Step 4, choose t_k as the largest $t \in J$ such that

$$F(\mathbf{x}^k + t\,\mathbf{v}^k) \leqq F(\mathbf{x}^k) + \beta t\, JF(\mathbf{x}^k)\mathbf{v}^k \leqq F(\mathbf{x}^k) - \beta t\,\mathbf{e},$$

where $t > 0$. Selecting t_k and restructuring the previous inequality, we get

$$F(\mathbf{x}^{k+1}) - F(\mathbf{x}^k) \leqq -\beta t_k\,\mathbf{e} < \mathbf{0}, \tag{5}$$

where $\mathbf{x}^{k+1} = \mathbf{x}^k + t_k\,\mathbf{v}^k$ from Step 5 of Algorithm 4.1. Thus we proved (4). □

In the following statement we show that the objective vector function values sequence $\{F(\mathbf{x}^k)\}$ generated by Algorithm 4.1 is $\mathbb{R}^m_{\oplus}$-decreasing.

Theorem 4.2. *Let $(GMOP)$ be given with $\mathcal{F} = \mathbb{R}^n$, and the vector function $F : \mathbb{R}^n \to \mathbb{R}^m$ continuously differentiable (and convex). Assume that Algorithm 4.1 starting at $\mathbf{x}^0 \in \mathcal{F}$ produces for $0 < \beta < 1$ the sequence of points $\{\mathbf{x}^k\} \subseteq \mathbb{R}^n$. Then, every accumulation point of $\{\mathbf{x}^k\}$, if any, is a substationary (weakly Pareto efficient) point to $(GMOP)$.*

Proof. Let $\bar{\mathbf{x}}$ be an accumulation point of the sequence $\{\mathbf{x}^k\}$. We should consider now a convergent subsequence of $\{\mathbf{x}^k\}$, but for simplicity we identify it with the whole sequence. Let $k \geq 0$. We have

$$\lim_{k\to\infty} \left(F\left(\mathbf{x}^{k+1}\right) - F\left(\mathbf{x}^k\right)\right) = F(\bar{\mathbf{x}}) - F(\bar{\mathbf{x}}) = 0 \tag{6}$$

and from (5) we get

$$F\left(\mathbf{x}^k\right) - F\left(\mathbf{x}^{k+1}\right) \geqq \beta t_k \mathbf{e} > 0. \tag{7}$$

From (6) and (7) follows that

$$\lim_{k\to\infty} t_k = 0. \tag{8}$$

On the other hand,

$$0 = \lim_{k\to\infty} \|\mathbf{x}^{k+1} - \mathbf{x}^k\| = \lim_{k\to\infty} \|\mathbf{x}^k + t_k\mathbf{v}^k - \mathbf{x}^k\| = \lim_{k\to\infty} \|t_k\mathbf{v}^k\|. \tag{9}$$

From (8) and (9) it follows that $\{\mathbf{v}^k\}$ is bounded. Hence, there exists an accumulation point $\bar{\mathbf{v}} \in \mathbb{R}^n$ of the sequence of joint decreasing directions $\{\mathbf{v}^k\}$ generated by Algorithm 4.1. Therefore, there exists a subsequence $\{\mathbf{v}^{k_j}\}$ such that

$$\lim_{k_j\to\infty} \mathbf{v}^{k_j} = \bar{\mathbf{v}}, \quad \text{and} \quad \lim_{k_j\to\infty} t_{k_j} = 0.$$

This means that for a fixed quite large $j_0 \in \mathbb{N}$ there exists a positive integer $\gamma > 1$ such that for all $j > j_0$ we have

$$t_{k_j} < \frac{1}{2^\gamma}, \tag{10}$$

which means that the inequality in Step 4 of the Algorithm 4.1 is not satisfied for $t = \frac{1}{2^\gamma}$, hence

$$F\left(\mathbf{x}^{k_j} + \frac{1}{2^\gamma}\mathbf{v}^{k_j}\right) \nleqq F\left(\mathbf{x}^{k_j}\right) + \frac{\beta}{2^\gamma} JF\left(\mathbf{x}^{k_j}\right)\mathbf{v}^{k_j}. \tag{11}$$

Thus, for each $j > j_0$ there exists $i = i(k_j) \in \{1, \ldots, m\}$ such that

$$F_i\left(\mathbf{x}^{k_j} + \frac{1}{2^\gamma}\mathbf{v}^{k_j}\right) > F_i\left(\mathbf{x}^{k_j}\right) + \frac{\beta}{2^\gamma}\nabla F_i\left(\mathbf{x}^{k_j}\right)^T\mathbf{v}^{k_j}. \tag{12}$$

Because the indices in $\{1, \ldots, m\}$ are finitely many, in the infinite sequence $\{k_j\}$ at least one of them, say i_0 should appear infinitely many times. The occurrence of $i_0 \in \{1, 2, \ldots, m\}$ defines the subsequence $\{k_{j_l}\}$. Then, for all $l = 1, 2, \ldots$ we have

$$F_{i_0}\left(\mathbf{x}^{k_{j_l}} + \frac{1}{2^\gamma}\mathbf{v}^{k_{j_l}}\right) > F_{i_0}\left(\mathbf{x}^{k_{j_l}}\right) + \frac{\beta}{2^\gamma}\nabla F_{i_0}\left(\mathbf{x}^{k_{j_l}}\right)^T\mathbf{v}^{k_{j_l}}. \tag{13}$$

We take the limit $l \to \infty$ and we get

$$F_{i_0}\left(\bar{\mathbf{x}} + \frac{1}{2^\gamma}\bar{\mathbf{v}}\right) \geq F_{i_0}(\bar{\mathbf{x}}) + \frac{\beta}{2^\gamma}\nabla F_{i_0}(\bar{\mathbf{x}})^T\bar{\mathbf{v}}. \tag{14}$$

This inequality holds for the positive integer γ defined by (10), and for i_0, hence $JF(\bar{\mathbf{x}})\bar{\mathbf{v}} \not< 0$, and it follows that

$$\max_{i=1,\ldots,m} \nabla F_i(\bar{\mathbf{x}})^T\bar{\mathbf{v}} \geq 0. \tag{15}$$

From (15) follows that there exists $\bar{\mathbf{w}} \in \mathbb{R}^m$ that solves (3) for the given $\bar{\mathbf{x}}$. Using Theorem 2.4 we get that $\bar{\mathbf{x}}$ is a substationary point to $(GMOP)$. Finally, in the convex case Remark 1 yields the assertion. □

In the following section we introduce a similar algorithm for solving MOPs with sign constraints.

5. Multiobjective optimization problems with sign constraints

Consider now the following MOPs with sign constraints

$$\left.\begin{matrix} \text{Min } F(\mathbf{x}) \\ \mathbf{x} \geqq \mathbf{0} \end{matrix}\right\} \qquad (SMOP).$$

Let the vector $\mathbf{x}=(\mathbf{x}_1,\dots,\mathbf{x}_n)^T \geqq \mathbf{0}$ be given. Then, we can define the following two sets of indices

$$I_+(\mathbf{x})=\{i\in\{1,\dots,n\} : \mathbf{x}_i>0\}, \quad \text{and} \quad I_0(\mathbf{x})=\{i\in\{1,\dots,n\} : \mathbf{x}_i=0\}, \tag{16}$$

denoted simpler by I_+ and I_0, respectively, when it is clear to which vector we refer. Using them we partition the columns of the matrix $JF(\mathbf{x})$ into two parts denoted by $JF(\mathbf{x})_{I_0}$ and $JF(\mathbf{x})_{I_+}$. Consider the linear programming problem

$$\left.\begin{array}{c}\max\limits_{(\mathbf{q},q_0)\in\mathbb{R}^n\times\mathbb{R}} q_0\\ JF(\mathbf{x})_{I_+}\mathbf{u}+JF(\mathbf{x})_{I_0}\mathbf{z}+q_0\,\mathbf{e}\leqq\mathbf{0}\\ 0\leq q_0\leq 1,\ \mathbf{z}\geqq\mathbf{0},\end{array}\right\} \qquad (LPS(\mathbf{x}))$$

where $\mathbf{q}^T=(\mathbf{u}^T,\mathbf{z}^T)$. The following statement is the counterpart of Theorem 2.4 for MOPs with sign constraints, exhibiting the connection between $(LPS(\mathbf{x}))$ and the substationarity of $\mathbf{x}\in\mathbb{R}^n_\oplus$ to $(SMOP)$ or the determination of a feasible joint decreasing direction of F on $\mathbb{R}^n_\oplus$.

Theorem 5.1 (*cf. Illés and Lovics (2018)*). *Let $(SMOP)$, $\mathbf{x}\geqq\mathbf{0}$ a feasible point of it and the associated $(LPS(\mathbf{x}))$ be given. Let the vector function $F:\mathbb{R}^n\to\mathbb{R}^m$ be continuously differentiable (and convex) on $\mathcal{F}=\mathbb{R}^n_\oplus$. Then, $(LPS(\mathbf{x}))$ always has an optimal solution with q_0^* and $\mathbf{q}^T=(\mathbf{u}^{*T},\mathbf{z}^{*T})$. There are two possibilities for the optimal value of $(LPS(\mathbf{x}))$, either $q_0^*=0$ which means that $\mathbf{x}$ is a substationary (weakly Pareto efficient) point to $(SMOP)$, or $q_0^*=1$ in which case $\mathbf{q}^T=(\mathbf{u}^{*T},\mathbf{z}^{*T})$ is a feasible joint decreasing direction of the vector function F on $\mathbb{R}^n_\oplus$.*

The new method for solving $(SMOP)$ is given in Algorithm 5.1. As mentioned later in Remark 8, we opted to construct the proposed algorithms gradually, instead of formulating first the one for the most complicate problem and then recovering those for simpler problems by removing constraints, to aid a better understanding of them.

Algorithm 5.1 Algorithm for solving sign constrained MOPs

[Step 1] take $\mathbf{x}^0\in\mathcal{F}=\mathbb{R}^n_\oplus$, $0<\beta<1$, $k=0$, $J=\left\{\frac{1}{2^n} : n\in\mathbb{N}\right\}$

[Step 2] determine the *(feasible) joint decreasing direction* $\mathbf{q}=\begin{pmatrix}\mathbf{u}^*\\ \mathbf{z}^*\end{pmatrix}=(q_1,\dots,q_n)^T\in\mathbb{R}^n$ from

$$\begin{array}{c}\max\limits_{(\mathbf{q},q_0)\in\mathbb{R}^n\times\mathbb{R}} q_0\\ JF(\mathbf{x}^k)_{I_+}\mathbf{u}+JF(\mathbf{x}^k)_{I_0}\mathbf{z}+q_0\,\mathbf{e}\leqq\mathbf{0}\\ 0\leq q_0\leq 1,\ \ \mathbf{z}\geqq\mathbf{0}.\end{array}$$

[Step 3] if $q_0=0$ then STOP ($\mathbf{x}^k$ is a substationary solution to $(SMOP)$);
otherwise compute $\sigma=\min\{\frac{\mathbf{x}_i^k}{-q_i} : q_i<0 \text{ and } i\in I_+\}$;
if $\sigma\geq 1$ then $\mathbf{v}^k:=\mathbf{q}$;
otherwise $\mathbf{v}^k:=\sigma\,\mathbf{q}$

[Step 4] choose t_k as the largest $t\in J$ s.t.

$$F(\mathbf{x}^k+t\,\mathbf{v}^k)\leqq F(\mathbf{x}^k)+\beta t\,JF(\mathbf{x}^k)\mathbf{v}^k$$

[Step 5] let $\mathbf{x}^{k+1}=\mathbf{x}^k+t_k\,\mathbf{v}^k$ and $k\leftarrow k+1$; go to Step 2

Lemma 5.2. *Let $(SMOP)$ be given with $\mathcal{F}=\mathbb{R}^n_\oplus$ and the vector function $F:\mathbb{R}^n\to\mathbb{R}^m$ be continuously differentiable on $\mathcal{F}$. Assume that Algorithm 5.1 starting at $\mathbf{x}^0\in\mathcal{F}$ produces the sequence of points $\{\mathbf{x}^k\}\subseteq\mathbb{R}^n_\oplus$. Then*

$$F(\mathbf{x}^{k+1})<F(\mathbf{x}^k), \qquad \forall k\geq 0. \tag{17}$$

Proof. Since $\mathbf{v}^k$ is a feasible joint decreasing direction computed in Step 2 of Algorithm 5.1, we have

$$JF(\mathbf{x}^k)_{I_+}\mathbf{u}^*+JF(\mathbf{x}^k)_{I_0}\mathbf{z}^*=JF(\mathbf{x}^k)\,\mathbf{v}^k\leqq-\mathbf{e}, \quad \text{and} \quad \mathbf{z}^*\geqq\mathbf{0}.$$

Therefore, in Step 4 of Algorithm 5.1 choose t_k as the largest $t\in J=\left\{\frac{1}{2^n} : n\in\mathbb{N}\right\}$ such that

$$F(\mathbf{x}^k+t\,\mathbf{v}^k)\leqq F(\mathbf{x}^k)+\beta t\,JF(\mathbf{x}^k)\mathbf{v}^k\leqq F(\mathbf{x}^k)-\beta t\,\mathbf{e},$$

and $\mathbf{x}^k+t\,\mathbf{v}^k\geqq\mathbf{0}$, where $t>0$. Selecting t_k and restructuring the previous inequality, we get

$$F(\mathbf{x}^{k+1})-F(\mathbf{x}^k)\leqq-\beta t_k\,\mathbf{e}<\mathbf{0}, \tag{18}$$

where $\mathbf{x}^{k+1}=\mathbf{x}^k+t_k\,\mathbf{v}^k$. Thus we proved (17). □

Theorem 5.3. *Let $(SMOP)$ be given with $\mathcal{F}=\mathbb{R}^n_\oplus$, and the vector function $F:\mathbb{R}^n\to\mathbb{R}^m$ continuously differentiable (and convex) on $\mathcal{F}$. Assume that Algorithm 5.1 starting at $\mathbf{x}^0\in\mathcal{F}$ produces the sequence of points $\{\mathbf{x}^k\}\subseteq\mathbb{R}^n_\oplus$. Then, every accumulation point of $\{\mathbf{x}^k\}$, if any, is a substationary (weakly Pareto efficient) point to $(SMOP)$.*

Proof. Let $\bar{\mathbf{x}}$ be an accumulation point of the sequence $\{\mathbf{x}^k\}$. The proof goes along the lines of the one of Theorem 4.2, one only needs to use Lemma 5.2 instead of Lemma 4.1, and then all the steps (6)–(13) hold.

Summarizing, the occurrence of $i_0\in\{1,2,\dots,m\}$ defines the subsequence $\{k_{j_l}\}$. Then, for all $l=1,2,\dots$ we have

$$F_{i_0}\left(\mathbf{x}^{k_{j_l}}+\frac{1}{2^\gamma}\mathbf{v}^{k_{j_l}}\right)>F_{i_0}\left(\mathbf{x}^{k_{j_l}}\right)+\frac{\beta}{2^\gamma}\nabla F_{i_0}\left(\mathbf{x}^{k_{j_l}}\right)^T\mathbf{v}^{k_{j_l}},$$

with the additional information that

$$\nabla F_{i_0}\left(\mathbf{x}^{k_{j_l}}\right)^T\mathbf{v}^{k_{j_l}}=\left[\nabla F_{i_0}\left(\mathbf{x}^{k_{j_l}}\right)\right]_{I_+}^T\mathbf{u}^{k_{j_l}}+\left[\nabla F_{i_0}\left(\mathbf{x}^{k_{j_l}}\right)\right]_{I_0}^T\mathbf{z}^{k_{j_l}},$$

where

$$I=I_+(\mathbf{x}^{k_{j_l}})\cup I_0(\mathbf{x}^{k_{j_l}}),\ \mathbf{v}^{k_{j_l}}|_{I_+}=\mathbf{u}^{k_{j_l}} \ \text{ and } \ \mathbf{v}^{k_{j_l}}|_{I_0}=\mathbf{z}^{k_{j_l}},\ \mathbf{z}^{k_{j_l}}\geqq\mathbf{0}.$$

In the infinite sequence of points $\{\mathbf{x}^{k_{j_l}}\}$ generated by Algorithm 5.1, there should be a partition $(\hat{I}_+,\hat{I}_0)$ of I that occurs infinitely many times. This is due to the fact that there are only finitely many possible different partitions of the index set I. Let us define the infinite index sequence $\iota(l)$ as a subsequence of k_{j_l} where the partition $(\hat{I}_+,\hat{I}_0)$ occurs. For $\mathbf{x}^{\iota(l)+1}=\mathbf{x}^{\iota(l)}+1/(2^\gamma)\mathbf{v}^{\iota(l)}$, we have

$$F_{i_0}\left(\mathbf{x}^{\iota(l)+1}\right)>F_{i_0}\left(\mathbf{x}^{\iota(l)}\right)+\frac{\beta}{2^\gamma}\left(\left[\nabla F_{i_0}\left(\mathbf{x}^{\iota(l)}\right)\right]_{\hat{I}_+}^T\mathbf{u}^{\iota(l)}+\left[\nabla F_{i_0}\left(\mathbf{x}^{\iota(l)}\right)\right]_{\hat{I}_0}^T\mathbf{z}^{\iota(l)}\right),$$

and $\mathbf{z}^{\iota(l)}\geqq\mathbf{0}$ for all indices $\iota(l)$. We take the limit $l\to\infty$ obtaining

$$\bar{\mathbf{x}}=\lim_{l\to\infty}\mathbf{x}^{\iota(l)},\ \ \bar{\mathbf{v}}=\lim_{l\to\infty}\mathbf{v}^{\iota(l)},\ \ \bar{\mathbf{u}}=\lim_{l\to\infty}\mathbf{u}^{\iota(l)},\ \ \bar{\mathbf{z}}=\lim_{l\to\infty}\mathbf{z}^{\iota(l)},\ \text{ with}$$

$$F_{i_0}\left(\bar{\mathbf{x}}+\frac{1}{2^\gamma}\bar{\mathbf{v}}\right)\geq F_{i_0}\left(\bar{\mathbf{x}}\right)+\frac{\beta}{2^\gamma}\left(\left[\nabla F_{i_0}\left(\bar{\mathbf{x}}\right)\right]_{I_+}^T\bar{\mathbf{u}}+\left[\nabla F_{i_0}\left(\bar{\mathbf{x}}\right)\right]_{I_0}^T\bar{\mathbf{z}}\right), \tag{19}$$

and $\bar{\mathbf{z}}\geqq\mathbf{0}$. Inequality (19) holds for the positive integer γ defined by (10), and for i_0, hence

$$JF(\bar{\mathbf{x}})\bar{\mathbf{v}}=JF(\bar{\mathbf{x}})_{I_+}\bar{\mathbf{u}}+JF(\bar{\mathbf{x}})_{I_0}\bar{\mathbf{z}}\not<\mathbf{0}, \quad \text{and} \quad \bar{\mathbf{z}}\geqq\mathbf{0}.$$

Therefore, the system of linear inequalities in $(\mathbf{u}^T,\mathbf{z}^T)^T\in\mathbb{R}^n$

$$JF(\bar{\mathbf{x}})_{I_+}\mathbf{u}+JF(\bar{\mathbf{x}})_{I_0}\mathbf{z}<\mathbf{0}, \quad \text{and} \quad \mathbf{z}\geqq\mathbf{0}$$

has no solution for the given feasible solution $\bar{\mathbf{x}}\geqq\mathbf{0}$. Applying Theorem 5.1, we get that $\bar{\mathbf{x}}$ is a substationary (weakly Pareto efficient) point to $(SMOP)$. □

6. Linearly constrained multiobjective optimization problem

In this section we consider linearly constrained MOPs of the form

$$\left.\begin{array}{l}\text{Min } F(\mathbf{x})\\ A\mathbf{x}=\mathbf{b}\\ \mathbf{x}\geqq\mathbf{0}\end{array}\right\} \qquad (LMOP),$$

where $A \in \mathbb{R}^{m\times n}$ is a matrix of rank m ($m \leq n$) and $\mathbf{b} \in \mathbb{R}^m$. Denote the *feasible set* of $(LMOP)$ by

$$\mathcal{P} = \{\mathbf{x} \in \mathbb{R}^n_{\oplus} : A\mathbf{x} = \mathbf{b}\}$$

and assume it nonempty. Like in Section 5, given $\mathbf{x}\geqq\mathbf{0}$, we define the index sets $I_+(\mathbf{x})$ and $I_0(\mathbf{x})$ exactly in the same way as in (16), used for partitioning the columns of the matrix $JF(\mathbf{x})$ into two parts denoted by $JF(\mathbf{x})_{I_0}$ and $JF(\mathbf{x})_{I_+}$, respectively. Using them, we define a linear programming problem that will be employed for deriving feasible joint decreasing directions to a generalized linearly constrained MOP at a point $\mathbf{x} \in \mathcal{P}$, namely

$$\left.\begin{array}{c}\max\limits_{(\mathbf{q},q_0)\in\mathbb{R}^n\times\mathbb{R}} q_0\\ JF(\mathbf{x})_{I_+}\mathbf{u} + JF(\mathbf{x})_{I_0}\mathbf{z} + q_0\,\mathbf{e}\leqq\mathbf{0}\\ A_{I_+}\mathbf{u} + A_{I_0}\mathbf{z} = \mathbf{0},\\ 0 \leq q_0 \leq 1,\ \mathbf{z}\geqq\mathbf{0}.\end{array}\right\} \quad (LPL(\mathbf{x}))$$

When $\mathbf{x} \in \mathcal{P}$, the first constraint of $(LPL(\mathbf{x}))$ guarantees that a solution $(\mathbf{u}^T, \mathbf{z}^T)^T \in \mathbb{R}^n$ defines in case $q_0 = 1$ a joint decreasing direction at $\mathbf{x}$ for the objective vector function F. The second and the third constraints of $(LPL(\mathbf{x}))$ guarantee the feasibility of the joint decreasing direction at $\mathbf{x}$ for problem $(LMOP)$. The following statement is the counterpart of Theorems 2.4 and 5.1 for MOPs with linear constraints, and, since it contains several modifications with respect to them, we provide its proof as well. The first occurrence of a similar result has been published in Illés and Lovics (2018), however, the proof given below slightly differs from the one available there.

Theorem 6.1. *Let $(LMOP)$ with a feasible point $\mathbf{x} \in \mathcal{P}$ and the associated $(LPL(\mathbf{x}))$ be given. Let the vector function $F : \mathbb{R}^n \to \mathbb{R}^m$ be continuously differentiable (and convex) on $\mathcal{P} = \mathcal{P}$. Assume that $\mathcal{P}$ is bounded. Then, $(LPL(\mathbf{x}))$ always has an optimal solution with q_0^* and $\mathbf{q}^T = (\mathbf{u}^{*T}, \mathbf{z}^{*T})$. There are two possibilities for the optimal value of $(LPL(\mathbf{x}))$, either $q_0^* = 0$ which means that $\mathbf{x}$ is a substationary (weakly Pareto efficient) point to $(LMOP)$, or $q_0^* = 1$ which means that $\mathbf{q}^T = (\mathbf{u}^{*T}, \mathbf{z}^{*T})$ is a feasible joint decreasing direction of the vector function F on $\mathcal{P}$.*

Proof. Note that $\mathbf{u} = \mathbf{0}$, $\mathbf{z} = \mathbf{0}$ and $q_0 = 0$ form a feasible solution to $(LPL(\mathbf{x}))$. Therefore, $(LPL(\mathbf{x}))$ has optimal solutions since it has feasible solutions and its objective function is bounded from above by 1.

Let us examine the case when the optimal objective function value of $(LPL(\mathbf{x}))$ is positive, namely $q_0 > 0$, thus the linear system of inequalities

$$\left.\begin{array}{c}JF(\mathbf{x})_{I_+}\mathbf{u} + JF(\mathbf{x})_{I_0}\mathbf{z} + q_0\,\mathbf{e}\leqq\mathbf{0}\\ A_{I_+}\mathbf{u} + A_{I_0}\mathbf{z} = \mathbf{0}\\ \mathbf{z}\geqq\mathbf{0},\ \ q_0 > 0\end{array}\right\} \quad (\mathcal{E}_1')$$

has solutions. This means that the system

$$\left.\begin{array}{c}JF(\mathbf{x})_{I_+}\mathbf{u} + JF(\mathbf{x})_{I_0}\mathbf{z} \leqq -\mathbf{e}\\ A_{I_+}\mathbf{u} + A_{I_0}\mathbf{z} = \mathbf{0},\ \ \mathbf{z}\geqq\mathbf{0}\end{array}\right\} \quad (\mathcal{E}_1)$$

has solutions as well. Any solution of $(\mathcal{E}_1)$ is a feasible joint decreasing direction to $(LMOP)$.

In case when the optimal objective function value of $(LPL(\mathbf{x}))$ is zero, namely $q_0 = 0$, then $(\mathcal{E}_1')$ has no solution, likewise $(\mathcal{E}_1)$ has no solution, either. According to the *Farkas Lemma*, the linear system of inequalities

$$\left.\begin{array}{c}\mathbf{w}^T JF(\mathbf{x})_{I_+} - \mathbf{v}^T A_{I_+} = \mathbf{0}\\ \mathbf{w}^T JF(\mathbf{x})_{I_0} - \mathbf{v}^T A_{I_0}\geqq\mathbf{0}\\ \mathbf{e}^T\mathbf{w} = 1,\ \ \mathbf{w}\geqq\mathbf{0}\end{array}\right\} \quad (\mathcal{E}_2)$$

has solutions. Let us analyze the weighted scalar optimization problem associated to $(LMOP)$, given as follows

$$\left.\begin{array}{l}\min\ \mathbf{w}^T F(\mathbf{x})\\ A\mathbf{x} = \mathbf{b}\\ \mathbf{x}\geqq\mathbf{0}\end{array}\right\} \quad (LMOP_{\mathbf{w}}),$$

with the weight $\mathbf{w} \in S_m$ that is part of a solution of $(\mathcal{E}_2)$. As $\mathcal{P}$ is nonempty, $(LMOP_{\mathbf{w}})$ has optimal solutions, that can be characterized by the KKT-conditions corresponding to $(LMOP_{\mathbf{w}})$, which are described in $(\mathcal{E}_2)$. Thus, any solution of $(\mathcal{E}_2)$ is a certificate that $\mathbf{x}$ is a substationary point to $(LMOP)$. □

The new iterative method we propose for solving $(LMOP)$ is given in Algorithm 6.1.

Algorithm 6.1 Algorithm for solving linearly constrained MOPs

[Step 1] take $\mathbf{x}^0 \in \mathcal{F} = \mathcal{P}$, $0 < \beta < 1$, $k = 0$, $J = \left\{\frac{1}{2^n} : n \in \mathbb{N}\right\}$

[Step 2] determine the *(feasible) joint decreasing direction* $\mathbf{q} = \begin{pmatrix}\mathbf{u}^*\\ \mathbf{z}^*\end{pmatrix} \in \mathbb{R}^n$ from

$$\begin{array}{c}\max\limits_{(\mathbf{q},q_0)\in\mathbb{R}^n\times\mathbb{R}} q_0\\ JF(\mathbf{x}^k)_{I_+}\mathbf{u} + JF(\mathbf{x}^k)_{I_0}\mathbf{z} + q_0\,\mathbf{e}\leqq\mathbf{0}\\ A_{I_+}\mathbf{u} + A_{I_0}\mathbf{z} = \mathbf{0}\\ 0 \leq q_0 \leq 1,\ \mathbf{z}\geqq\mathbf{0}\end{array}$$

[Step 3] if $q_0 = 0$ then STOP ($\mathbf{x}^k$ is a substationary solution to $(LMOP)$);
otherwise compute $\sigma = \min\{\frac{\mathbf{x}_i^k}{-q_i} : q_i < 0 \text{ and } i \in I_+\}$;
if $\sigma \geq 1$ then $\mathbf{v}^k := \mathbf{q}$;
otherwise $\mathbf{v}^k := \sigma\,\mathbf{q}$;

[Step 4] choose t_k as the largest $t \in J$ s.t.

$$F(\mathbf{x}^k + t\,\mathbf{v}^k) \leqq F(\mathbf{x}^k) + \beta t\, JF(\mathbf{x}^k)\mathbf{v}^k$$

[Step 5] let $\mathbf{x}^{k+1} = \mathbf{x}^k + t_k\,\mathbf{v}^k$ and $k \leftarrow k + 1$; go to Step 2

Remark 6. The sequence $\{\mathbf{x}^k\}$ generated in Algorithm 6.1 is feasible to $(LMOP)$ because $\mathbf{x}^0 \in \mathcal{P}$ and, given $k \in \mathbb{N}$, when $\mathbf{x}^k \in \mathcal{P}$ one has, by construction, $\mathbf{x}^{k+1} \in \mathbb{R}^n_{\oplus}$ (due to scaling the joint decreasing direction with σ in Step 3) and $A\mathbf{x}^{k+1} = A\mathbf{x}^k + t_k A\mathbf{v}^k = \mathbf{b} + \mathbf{0} = \mathbf{b}$. From a practical point of view, finding initial starting points for the linear subproblems of Algorithm 6.1 was intensively discussed in the theory and applications of linear programming (see, for instance, Maros, 2003; Murty, 1976; Roos et al., 2005). For portfolio optimization problems, we used the modeling language Mosel in FICO Xpress Optimizer to implement Algorithm 6.1 and took all the advantages of this state-of-the-art solver in our implementation, like determining extremely fast optimal solutions to the linear programming subproblems (see Section 7).

In the following statement we prove that the objective vector function values sequence $\{F(\mathbf{x}^k)\}$ is $\mathbb{R}^m_{\oplus}$-decreasing.

Lemma 6.2. *Let $(LMOP)$ be given with $\mathcal{F} = \mathcal{P} \neq \emptyset$ and $\mathcal{P}$ is assumed to be bounded. Furthermore, let the vector function $F : \mathbb{R}^n \to \mathbb{R}^m$ be continuously differentiable on $\mathcal{F}$. Assuming that Algorithm 6.1 starting at $\mathbf{x}^0 \in \mathcal{F}$ produces the sequence of points $\{\mathbf{x}^k\}$, then*

$$F(\mathbf{x}^{k+1}) < F(\mathbf{x}^k), \qquad \forall k \geqq 0. \tag{20}$$

Proof. Since $\mathbf{v}^k$ is a feasible joint decreasing direction computed in Step 2 of Algorithm 6.1 using the linear programming problem $(LPL(\mathbf{x}^k))$, the proof goes along the same lines as the one of Lemma 5.2. □

Theorem 6.3. *Let $(LMOP)$ be given with $\mathcal{F} = \mathcal{P}$ such that $\mathcal{P} \neq \emptyset$ and it is bounded. Furthermore, the vector function $F : \mathbb{R}^n \to \mathbb{R}^m$ is continuously differentiable (and convex) on $\mathcal{F}$. Assume that Algorithm 6.1 starting at $\mathbf{x}^0 \in \mathcal{F}$ produces the sequence of points $\{\mathbf{x}^k\}$. Then, every accumulation point of $\{\mathbf{x}^k\}$, if any, is a substationary (weakly Pareto efficient) point to $(LMOP)$.*

Proof. Let $\bar{\mathbf{x}}$ be an accumulation point of the sequence $\{\mathbf{x}^k\}$. The proof goes along the same lines as the one of Theorem 5.3, as the difference in the linear programming problems considered in Step 2 of the corresponding algorithms does not affect its flow, since the additional constraint only ensures the feasibility of the new iterates after applying the computed joint decreasing direction. Only at the very end of the proof we need to handle the question of the properties of $\bar{\mathbf{x}}$.

Like in the proof of Theorem 5.3, the system of linear inequalities in $(\mathbf{u}, \mathbf{z}) \in \mathbb{R}^n$

$$JF(\bar{\mathbf{x}})_{I_+}\mathbf{u} + JF(\bar{\mathbf{x}})_{I_0}\mathbf{z} < \mathbf{0}, \quad \text{and} \quad \mathbf{z} \geqq \mathbf{0}$$

has no solution for the given feasible solution $\bar{\mathbf{x}} \in \mathcal{P}$. Therefore, the linear system of inequalities $(\mathcal{E}_1)$ has no solution, either. Exactly the same steps as in the proof of Theorem 6.1 lead to the same conclusion, that $\bar{\mathbf{x}} \in \mathcal{P}$ is a substationary (weakly Pareto efficient) point to $(LMOP)$. □

Remark 7. The fact that $\mathcal{P}$ is assumed to be bounded was not used directly in the proof of Theorem 6.1 and in the one of Theorem 6.3. In both cases, we imposed this assumption to guarantee that $(LMOP_{\mathbf{w}})$ has optimal solutions. This can be ensured with different assumptions as well, that would be discussed in a subsequent work. In particular, when F is convex on $\mathcal{P}$, $(LMOP_{\mathbf{w}})$ is a convex optimization problem and thus each solution of $(\mathcal{E}_2)$ (their existence is guaranteed by the *Farkas Lemma* whenever Algorithm 6.1 stops) delivers an optimal solution to $(LMOP_{\mathbf{w}})$, making thus the assumption of boundedness on $\mathcal{P}$ not necessary.

Remark 8. Algorithm 4.1 can be derived as a special case of Algorithm 5.1, which itself is a particular instance of Algorithm 6.1. However, presenting each of them in the appropriate framework allows one note how adding constraints (that complicate the considered problem) to the initial unconstrained MOP (first sign ones, then also linear ones) leads to corresponding modifications in the algorithms, too.

7. Numerical results

We implemented Algorithms 4.1 and 6.1 in MATLAB in order to showcase the practical usage of the iterative methods we propose in this work. We did all the computations on a desktop computer with Intel(R) Core(TM) i5-8250U CPU, 1.60 GHz processor and 8 GB RAM. For the larger size portfolio optimization problems with real life data, Algorithm 6.1 has been implemented in the modeling language MOSEL in the state-of-the-art commercial solver FICO XPRESS OPTIMIZER.

Except for the conditional gradient method proposed in Assunção et al. (2021) and W. Chen et al. (2023) that is only able to solve MOPs with compact constraint sets, none of the algorithms known so far for solving MOPs (see Section 3) is fully implementable without resorting to advanced external numerical solvers in order to deal with the intermediate subproblems they contain. By *efficient advanced solvers* we mean software products which are as efficient and precise as they could be from a numerical point of view. None of the MATLAB-based solvers (e.g. FMINCON) is actually efficient in this sense. For numerical problems of medium and large size, MATLAB is not really a good option, and *state-of-the-art* advanced optimization solvers need to be used for solving instances of the well-known portfolio optimization problems, formulated as MOPs. A representative of such solvers is FICO XPRESS OPTIMIZER, that we employ for solving a portfolio MOP. The implementations are made in (the programming language) MOSEL, and the joint decreasing direction subproblems (which are linear) were solved by the linear programming solver of FICO XPRESS OPTIMIZER. However, as one can see below, for small size test problems, the algorithms we propose perform well even in MATLAB, unlike their counterparts from the literature, that rely on employing functions defined in MATLAB. It is clear that implementations of algorithms for MOPs in MATLAB are much slower for medium and large size problem than those using state-of-the-art solvers like FICO XPRESS OPTIMIZER.

As already mentioned, in case of constrained MOPs, for the iterative methods from the literature (Algorithms 3.5 and 3.6) it is not a trivial task to find feasible descent directions, since projections onto feasible solution sets are incorporated in the computation of feasible descent directions. On the other hand, the feasibility of the joint decreasing directions generated in Algorithms 5.1 and 6.1 is automatically guaranteed by construction.

In Section 7.1 our goal is to solve small-sized test MOPs where we can illustrate the progress of the computational procedures. In Example 2 we modify Example 1 by adding linear constraints to show their effect on the Pareto optimal solution set. In Section 7.2 we deal with portfolio optimization problems taking into consideration up to 50 shares from *Euro Stoxx 50* given in Vörös et al. (2026), Vörös and Rappai (2025), where the authors considered data from the period 01.01.2024–01.01.2025 and computed the covariance matrix of the shares and the coefficients of the expected value function (that is linear). Using these data, we build several test problems for which we show the fast computational performance of Algorithm 6.1.

Direct comparisons of the performance of the algorithms we propose with others from the literature are beyond the scope of this work, while full and efficient implementations of our methods (e.g. in the sense suggested in Remark 5) will be the object of subsequent investigations.

7.1. Different aspects of MOPs

The main scope of the numerical experiments presented below is to show that the methods we propose are indeed implementable and deliver what they are expected to, namely substationary or weakly Pareto efficient points to the considered test MOPs. In MATLAB we used the built-in function LINPROG for solving the intermediate linear programming problems, while FICO XPRESS OPTIMIZER has its own incorporated linear programming solver. Note that we always have a starting feasible solution for the linear programming problems defined in Step 2 of the considered algorithms, e.g. $(\mathbf{0}, 0)$, and, as mentioned in Remark 5, whenever $q_0 > \zeta$ with $\zeta \in (0.1, 1)$, and $\mathbf{q} \neq \mathbf{0}$ we can stop the linear programming solver, as a joint decreasing direction has been generated. From the test examples available in the literature (see, for instance, Ansary and Panda (2015)) we chose a convex unconstrained MOP to show how our algorithm performs. After that, we modified the selected test problem by adding linear constraints in such a way, that some of the weakly Pareto efficient points became infeasible for the constrained problem. Our numerical experiments using different starting points show the effect of having linear constraints, as well, see Figs. 4–6.

Example 1. Let $F_1, F_2, F_3 : \mathbb{R}^2 \to \mathbb{R}$ be defined as follows

$$F_1(\mathbf{x}) = \frac{1}{4}\left((x_1 - 1)^4 + 2(x_2 - 2)^4\right), \; F_2(\mathbf{x}) = e^{\frac{x_1 + x_2}{2}} + x_1^2 + x_2^2,$$
$$F_3(\mathbf{x}) = \frac{1}{6}\left(e^{-x_1} + 2\,e^{-x_2}\right)$$

and let $F : \mathbb{R}^2 \to \mathbb{R}^3$, $F(\mathbf{x}) = (F_1(\mathbf{x}), F_2(\mathbf{x}), F_3(\mathbf{x}))^T$. We want to solve the following convex unconstrained MOP

$$\left.\begin{array}{l} \text{Min } F(\mathbf{x}) \\ \mathbf{x} \in \mathbb{R}^2. \end{array}\right\}$$

We used two types of stopping criteria in our implementation, namely we ran the algorithm until

i. $\|t_k\mathbf{q}\| \leq \epsilon$ or $q_0 \leq \epsilon$, or
ii. $t_k \leq \epsilon$ or $q_0 \leq \epsilon$,

where t_k is the stepsize and $(\mathbf{q}, q_0)$ is the derived optimal solution to the linear programming problem from Step 2 of Algorithm 4.1. While the stopping criterion $q_0 \leq \epsilon$ is an obvious choice from a numerical point of view, e.g. when $\epsilon \in (0, 10^{-5})$, the other two criteria (namely $\|t_k\mathbf{q}\| \leq \epsilon$ and $t_k \leq \epsilon$) are justified by computational reasons. Both of them ensure that the iterative process stops when the advances become insignificant. Furthermore, we also stopped the algorithm when the number of iterations reached 500. In Figs. 1 and 2 we can see the obtained results with different values of $\epsilon > 0$ by considering the starting point $(-1, 2)^T$ and $(0, 3)^T$, respectively. In the first two cases we used the stopping criterion *i.*, while in the third one we applied the stopping criterion *ii*. The obtained weakly Pareto efficient points are the same within the limits of numerical errors. From the obtained numbers of iterations we can see that the algorithm is sensitive to the employed stopping criterion and to the admissible error $\epsilon > 0$. Note that we used in our computational experiments various values of the parameter $\beta \in (0, 1)$, as "indicated" by the performance of the employed algorithms in order to optimize their performance. For instance, the value $\beta = 0.1$ was chosen both because of "historical" reasons (as the similar algorithms in the literature use it) and also to provide additional flexibility to our methods. It is known from the nonlinear optimization literature that β, known as the *Armijo parameter*, cannot be dropped, however, its optimal value is not known in general. Taking this into consideration, we even did an experiment to evaluate the effect of the value of β on the efficiency of the solution process for given portfolio optimization MOPs. However, there is no evidence of a certain "monotonicity" of the elapsed time as β increases and the computational results do not indicate anything in this sense.

In order to get an idea about (a subset of) the whole set of weakly Pareto efficient points of the considered MOP, we provide a so-called *inner approximation* of it, that comprises the outcomes of starting the proposed algorithm from as many as possible points (not to be confused with the *outer approximation* of a solution set of a MOP, for which other methods are available, see Illés and Lovics (2018), also Wu and Yang (2025) for a branch and bound approach). To proceed, we divided the rectangle $\mathcal{T} = [-2, 8] \times [-2, 8]$ into 10×10 unit squares and we generated 5 points in each square. In this way, we obtained 500 starting points in $\mathcal{T}$. We ran Algorithm 4.1 with these starting points with $\epsilon = 10^{-5}$ and by using the stopping criterion *i*. In Fig. 3 we can see the obtained weakly Pareto efficient points after running the algorithm 500 times with these starting points. Hence, Fig. 3 shows how a portion of the the set of weakly Pareto efficient points looks like in case of Example 1. Moreover, this experiment shows that different starting points lead to different weakly Pareto efficient points. Understanding the way inputs and outcomes of our algorithm are connected could be the topic of a subsequent paper.

We also tested our new method on a linearly constrained MOP.

Example 2. Consider the following problem

$$\left.\begin{array}{l} \text{Min } F(\mathbf{x}) \\ A\mathbf{x} \leqq \mathbf{b} \\ \mathbf{x} \geqq \hat{\mathbf{x}}, \end{array}\right\}$$

where F is the function given in Example 1, $A = \begin{pmatrix} 10 & 1 \\ 0 & 1 \\ -1 & -1 \end{pmatrix}$, $\mathbf{b} = \begin{pmatrix} 8 \\ 5 \\ 0 \end{pmatrix}$ and $\hat{\mathbf{x}} = \begin{pmatrix} -2 \\ -2 \end{pmatrix}$.

Note that this problem is slightly different to $(LMOP)$ in the sense that instead of $A\mathbf{x} = \mathbf{b}$ we have $A\mathbf{x} \leqq \mathbf{b}$ and also a different lower bound imposed on the vector $\mathbf{x}$. To accommodate this situation, in our numerical computations we modified Step 2 of Algorithm 5.1. The

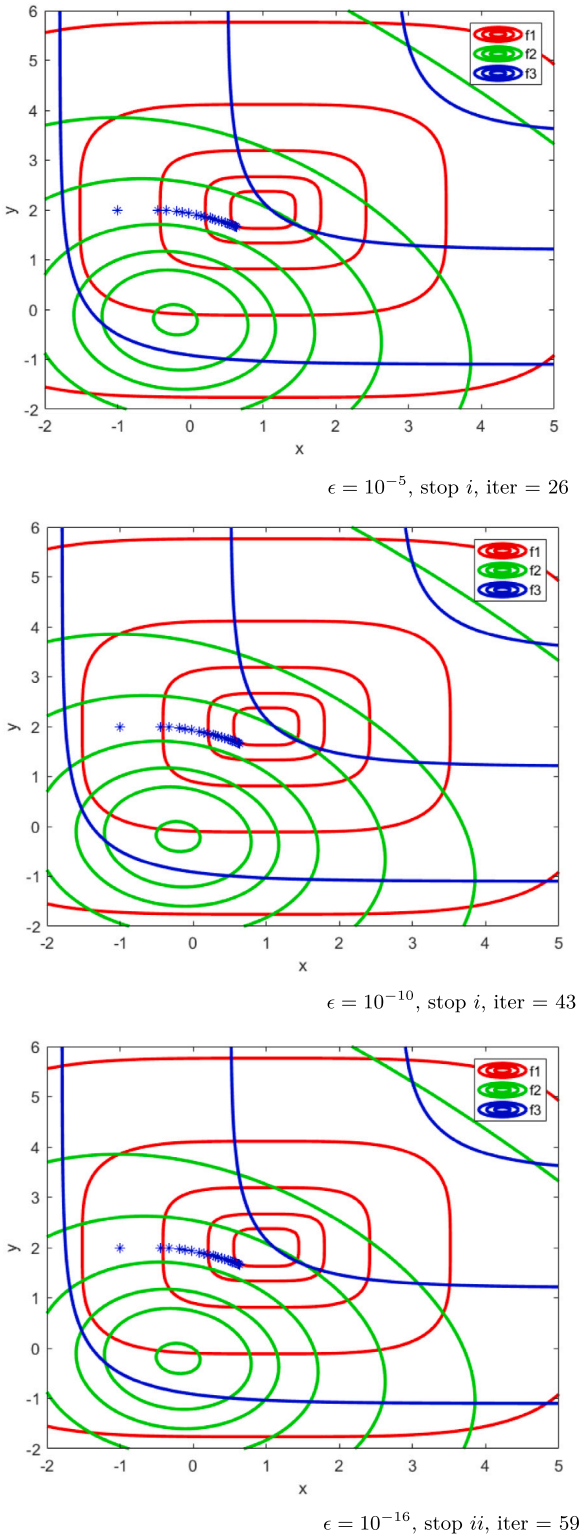
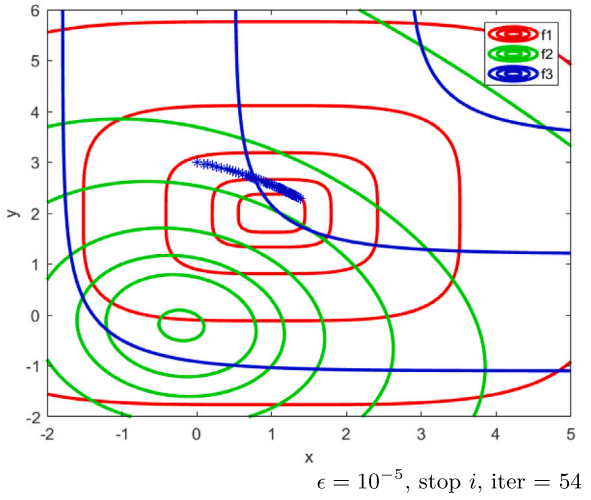
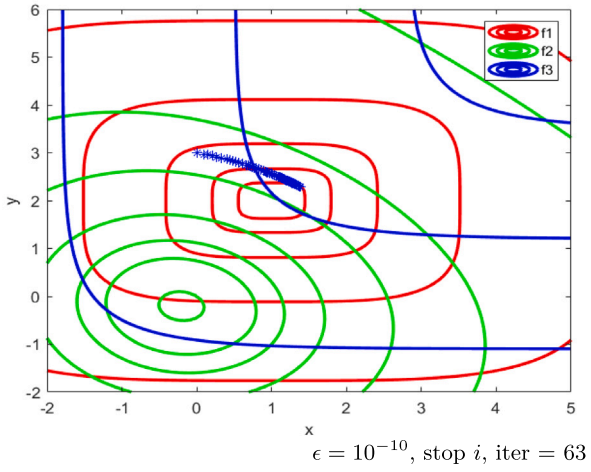


Fig. 1. Results with starting point $(-1, 2)^T$. The level sets of the used objective functions F_1, F_2 and F_3 are plotted in red, green and blue, respectively. The blue dots represent the iterates, i.e. the values of $\{\mathbf{x}^k\}$ until the stopping criterion is activated. Here, "stop *i*" or "stop *ii*" represents the stopping criterion *i* or *ii*, while "iter" is the number of performed iterations before the activation of the stopping criterion. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)





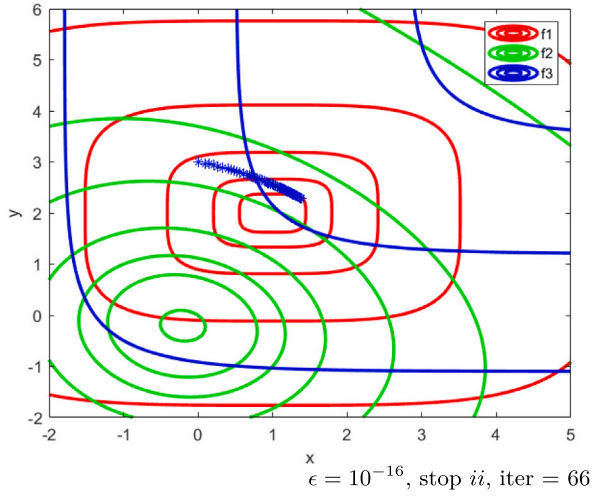


Fig. 2. Results with starting point $(0, 3)^T$. The level sets of the used objective functions F_1, F_2 and F_3 are plotted in red, green and blue, respectively. The blue dots represent the iterates, i.e. the values of $\{\mathbf{x}^k\}$ until the stopping criterion is activated. Here, "stop i" or "stop ii" represents the stopping criterion i or ii, while "iter" is the number of performed iterations before the activation of the stopping criterion. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

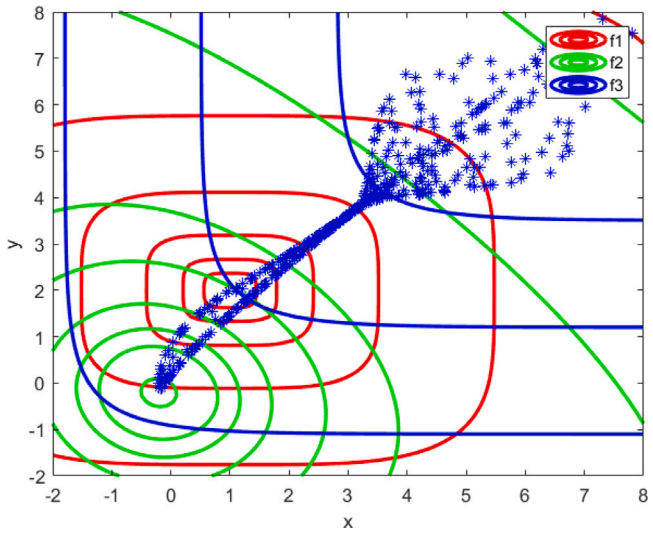


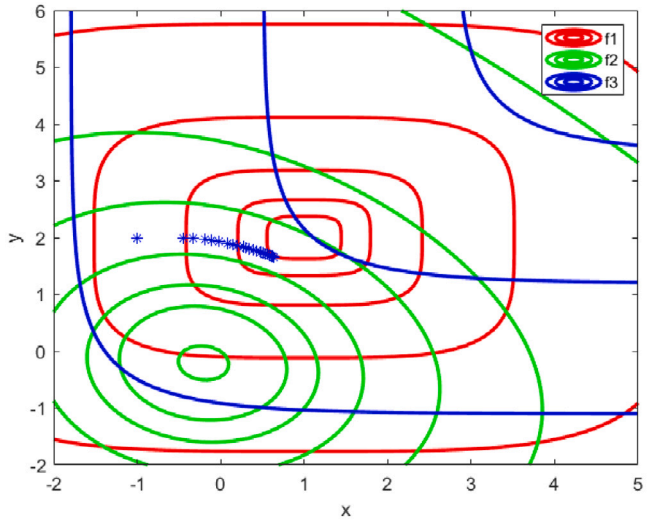
Fig. 3. The approximation of the set of weakly Pareto efficient points obtained by means of 500 random starting points in $[-2, 8] \times [-2, 8]$.

obtained results can be seen in Figs. 4 and 5, where we compared the results with the unconstrained case. In all these cases we used $\beta = 0.1$, $\epsilon = 10^{-5}$ and stopping criterion i. In the captions of the figures we can see the obtained weakly Pareto efficient point within the limits of the numerical calculation errors and the necessary numbers of iterations. From Figs. 4 and 5 we can see the effect of the linear constraints on the solution. In Fig. 4, when we have linear constraints, the algorithm stops with q_0 becoming less than ϵ. We can see that the obtained weakly Pareto efficient point and the trajectory also differs from the unconstrained case, because the point $\bar{\mathbf{x}}_u$ obtained in the unconstrained case is not feasible to the constrained problem. In Fig. 5, when we have linear constraints, the obtained weakly Pareto efficient point is on the boundary of the feasible polyhedron, and the followed trajectory is also different than in the unconstrained case. In both cases the numbers of iterations are also different, because the obtained point $\bar{\mathbf{x}}_u$ is not feasible to the constrained problem. Moreover, Fig. 5 helps us to understand that starting from any feasible point our algorithm delivers a weakly Pareto efficient point that may differ from the corresponding outcome in the unconstrained case, even if the latter is feasible to the considered constrained MOP.

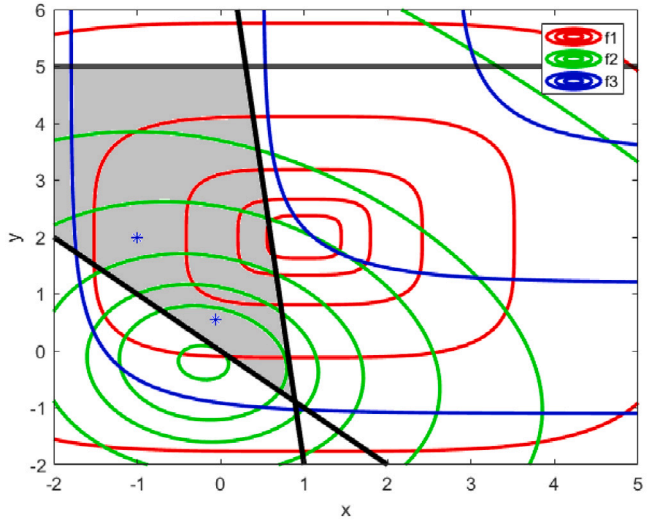
Let us define the rectangle $\mathcal{T} = [-2, 1] \times [-1, 5]$, such that it contains the feasible solution set of the problem. We divided $\mathcal{T}$ into 10×10 unit squares and we intersected the squares with the polyhedron obtained from the linear constraints coming from Example 2. We generated 10 points in each obtained square. We ran Algorithm 6.1 with these feasible starting points with $\beta = 0.1$, $\epsilon = 10^{-5}$ and by using the stopping criterion i. In Fig. 6 we can see the obtained weakly Pareto efficient points in each case. We can see that in many cases the algorithm stopped at the boundary of the polyhedron, which are weakly Pareto efficient points for the constrained case. Based on our observation, the running time in this case was 399 s, because we have less starting points and the linear constraints, as well.

7.2. Portfolio optimization problems

In the following we report on employing Algorithm 6.1 for solving portfolio optimization problems using real stock market data. The *EURO Stoxx 50* shares data for the period of 01.01.2024. – 01.01.2025 have been collected in Vörös et al. (2026), Vörös and Rappai (2025), who computed the expected value coefficients for each share, and the corresponding covariance matrix as well.

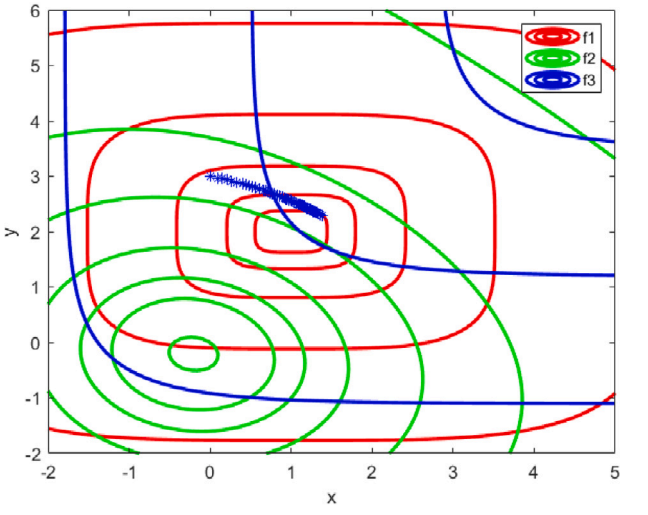


$\bar{\mathbf{x}}_{\mathbf{u}} = (0.6407, 1.6583)^T$, iter $= 26$

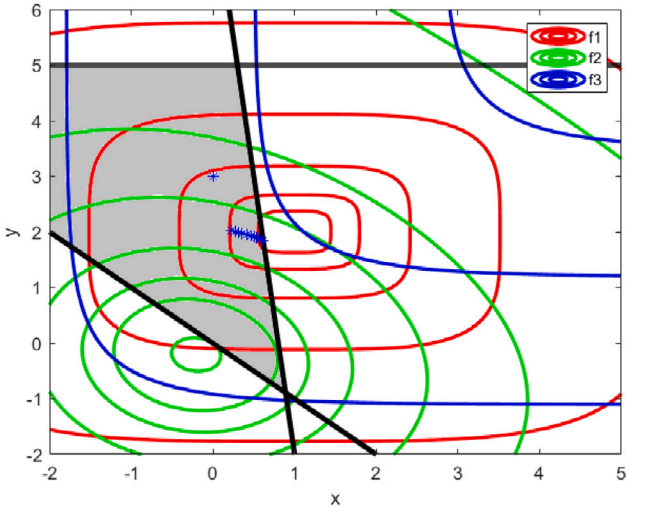


$\bar{\mathbf{x}}_{\mathbf{c}} = (-0.0556, 0.5556)^T$, iter $= 2$

Fig. 4. Results with starting point $(-1, 2)^T$ without constraints and with linear constraints. The lines representing the linear constraints are plotted in black. The blue dots represent the iterates, i.e. the values of $\{\mathbf{x}^k\}$ until the stopping criterion is activated. Here, $\bar{\mathbf{x}}_{\mathbf{u}}$ and $\bar{\mathbf{x}}_{\mathbf{c}}$ represent the obtained weakly Pareto efficient points within the numerical tolerance, and "iter" is the number of performed iterations before the activation of the stopping criterion. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)



$\bar{\mathbf{x}}_{\mathbf{u}} = (1.3861, 2.2824)^T$, iter $= 54$



$\bar{\mathbf{x}}_{\mathbf{c}} = (0.6150, 1.8498)^T$, iter $= 11$

Fig. 5. Results with starting point $(0, 3)^T$ without constraints and with linear constraints. The lines representing the linear constraints are plotted in black. The blue dots represent the iterates, i.e. the values of $\{\mathbf{x}^k\}$ until the stopping criterion is activated. Here, $\bar{\mathbf{x}}_{\mathbf{u}}$ and $\bar{\mathbf{x}}_{\mathbf{c}}$ represent the obtained weakly Pareto efficient points within the numerical tolerance, and "iter" is the number of performed iterations before the activation of the stopping criterion. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Example 3. Consider the portfolio optimization problem (see Markowitz (1952, 1959))

$$\left.\begin{array}{l}\text{Min} \begin{pmatrix} -\mathbf{x}^T\mathbf{u} \\ \mathbf{x}^\top\mathbf{V}\mathbf{x} \end{pmatrix} \\ \mathbf{x} \in \mathcal{F},\end{array}\right\} \qquad (POP)$$

with $\mathcal{F} = \left\{\mathbf{x} \in \mathbb{R}^d_{\oplus} : \sum_{i=1}^d \mathbf{x}_i = 1\right\}$, where $\mathbf{u} \in \mathbb{R}^d$ and $\mathbf{V} \in \mathbb{R}^{d\times d}$ is a symmetric positive semidefinite matrix.

In portfolio optimization, $\mathbf{x}$ represents the portfolio vector for d given assets (i.e. the proportions of the assets in the whole portfolio), where the short sales are excluded, the first component of the objective vector function is the negative of the expected return, and the second one is the variance of the portfolio, that quantifies the risk associated to the considered portfolio.

From the *EURO Stoxx 50* shares data provided in Vörös et al. (2026), Vörös and Rappai (2025), by selecting the first 7, 14, 28 and the whole set of 50 shares (i.e. $d = 7, 14, 28, 50$), we built portfolio optimization problems of different dimensions, aiming to analyze the effect of the problem size on the running time of our algorithm. For each such portfolio optimization problem, we employed 10 different starting portfolios to find out how these influence the solution speed to a weakly Pareto efficient solution.

We implemented our Algorithm 6.1 for solving portfolio optimization problems in the Mosel modeling language in the FICO Xpress Optimizer solver (FICO, 2025). From a practical point of view, we had

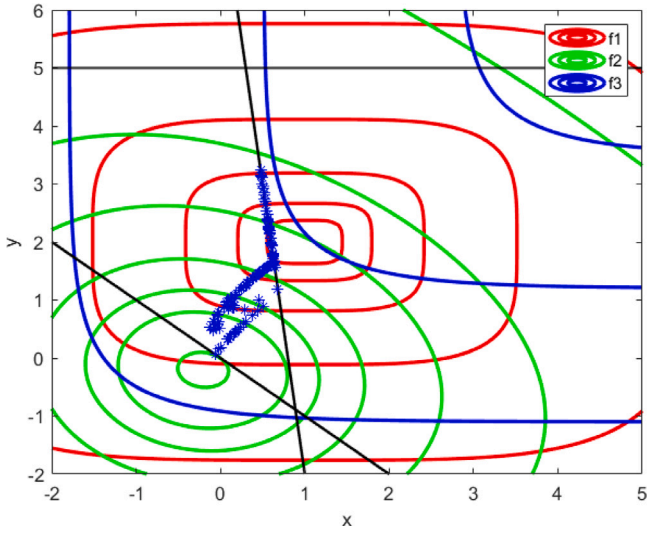


Fig. 6. The approximation of the set of weakly Pareto efficient points obtained by means of 1000 random starting points in $[-2,1]\times[-1,5]$.

Table 1
LP - number of linear programming problems solved to find feasible joint decreasing directions, I_{min}, I_{max} - minimum and maximum number of iterations for a given feasible joint decreasing direction to find the proper step length in Step 4, I_{tot} - total number of iterations for all feasible joint decreasing directions to find the proper step lengths in Step 4, time (s) - the total execution time of the algorithm in seconds, for $\beta = 0.99$.

Point	LP	I_{min}	I_{max}	I_{tot}	Time (s)
P_1	24	2	17	310	**0.175**
P_2	1	29	29	29	0.029
P_3	1	30	30	30	0.040
P_4	1	29	29	29	0.033
P_5	1	25	25	25	0.044
P_6	3	3	18	24	0.043
P_7	1	28	28	28	0.072
P_8	4	4	24	38	0.040
P_9	19	3	21	325	**0.255**
P_{10}	26	4	21	432	**0.126**

to modify the stopping criteria in two ways. If $t_k < \varepsilon$, then we used the most recently determined portfolio to find a new joint decreasing direction, and only in case the ℓ_2-norm distance between two consecutive computed portfolios is small enough, then we stopped the computation, declaring that we found an approximate weakly Pareto efficient point to the considered portfolio optimization problem.

Let us show first the computational results related to the portfolio optimization problem with $d = 14$, started from 10 different portfolios (see Table 1). In these experiments we used $\beta = 0.99$ (see Algorithm 6.1).

These tests (see Table 1) show that the starting portfolio influences: (i) how many times we need to compute joint decreasing directions (how many linear programming problems we need to solve), (ii) how many times we need to compute step length. Clearly, the running time of our algorithm depends on (i) and (ii). Starting portfolios P_1, P_9 and P_{10} show that the necessary number of linear programming problems to find an approximate weakly Pareto efficient solution deeply influences the running time of our algorithm. Therefore, it is very important to use solvers which solve linear programming problems very efficiently.

Algorithm 6.1 contains the important parameter $\beta \in (0,1)$. It is clear that the chosen value of β has influence on the step length t_k, namely a larger value of β may shorten, many times, the step length values. This could have a great impact on the performance of the Algorithm 6.1. For this reason we tested how the selected values of β (0.99, 0.75, 0.4, 0.1, 0.01) affect the solutions of different test problems (P_1, P_9 and P_{10}).

Table 2
This table shows for three portfolios P_1, P_9 and P_10 the effect of choosing different values for parameter $\beta \in (0,1)$. These influence not only the steplength at different joint dcreasing directions, but the corresponding number of linear programming subproblems that need to be solved, too. In both ways, β affects the total computational time of the Algorithm 6.1, as it is expected.

$\beta = 0.99$					
Point	LP	I_{min}	I_{max}	I_{tot}	Time (s)
P_1	24	2	17	310	0.175
P_9	19	3	21	325	0.255
P_{10}	26	4	21	432	0.126
$\beta = 0.75$					
Point	LP	I_{min}	I_{max}	I_{tot}	Time (s)
P_1	3	2	24	28	0.045
P_9	5	3	24	50	0.043
P_{10}	2	3	31	34	0.042
$\beta = 0.4$					
Point	LP	I_{min}	I_{max}	I_{tot}	Time (s)
P_1	2	1	40	41	0.044
P_9	3	3	31	39	0.068
P_{10}	2	3	31	34	0.068
$\beta = 0.1$					
Point	LP	I_{min}	I_{max}	I_{tot}	Time (s)
P_1	2	1	40	41	0.032
P_9	2	2	34	36	0.042
P_{10}	2	3	32	35	0.034
$\beta = 0.01$					
Point	LP	I_{min}	I_{max}	I_{tot}	Time (s)
P_1	2	1	40	41	0.058
P_9	2	2	34	36	0.043
P_{10}	2	3	32	35	0.044

Table 3
d-number of portfolios, β-parameter used in Step 4, LP_a - average number of solved linear programming problems in 10 runs, a_{min}, a_{max} - average of minimum and maximum number of iterations in 10 runs to find the proper step length in Step 4, a_{tot} - average of total computations of step lengths in 10 runs, $t_a(s)$- average time of 10 runs in seconds.

d	β	LP_a	a_{min}	a_{max}	a_{tot}	$t_a\,(s)$
7	0.75	9.2	5.6	22.4	132.6	0.068
14	0.99	8.1	15.7	24.2	127	0.086
28	0.1	3	4.5	30.1	44.6	0.053
50	0.4	14.9	1.3	26.4	313.2	0.185

From the computations it is clear that for smaller values of β fewer linear programming problems need to be solved (see Table 2). However, the running time does not only depend on the number of linear programming programming subproblems, because it could happen that decreasing β, the computation of the step length increases the total running time of the Algorithm 6.1. Finally, let us present in a single table (see Table 3) all 40 runs. This table displays the effect of the number of shares, the number of linear programming problems need to be solved, the number of times the step length needs to be computed and the selected value of β.

We may finalize our observations as follows: starting portfolios, number of shares in the portfolio, and selected value of β have great effect on the running time of our algorithm. Clearly, the major advantage of our algorithm is that for finding a (feasible) joint decreasing direction we need to solve (only) a linear programming problem. Both from theoretical and computational points of view it is known that solving linear programming problems is more efficient than solving convex nonlinear ones (Glineur, 2001; Maros, 2003; Nesterov, 2018; Roos et al., 2005). In the case of a large number of shares, this advantage of our algorithm would be even more conspicuous. Thus, working in

an environment that ensures fast solvability of linear programming problems (like FICO Xpress Optimizer, see FICO (2025)) is crucial for efficiently implementing Algorithms 4.1, 5.1, and 6.1.

8. Conclusions and further research plans

We proposed a generic scheme for solving MOPs that can be used for large-scale problems. In particular, we derived new algorithms for solving unconstrained, sign constrained and linearly constrained MOPs, where we computed the feasible joint decreasing direction using suitably defined linear programming problems. In all the cases we showed that the objective vector function values sequence is decreasing with respect to the considered orthant. We proved that every accumulation point of the sequence generated by the algorithms, if any, is a substationary point to the considered MOPs. When the objective vector function of the considered MOP is convex with respect to the corresponding nonnegative orthant or linear constraints, each accumulation point of the mentioned sequence turns out to be weakly Pareto efficient. Different to their existing counterparts from the literature, a novelty of these algorithms is that the decreasing directions are easily computable via linear programming, enabling our approach to solve larger-scaled problems. Moreover, while the existing algorithms require determining the unique optimal solution to a (constrained) minimax quadratic problem, our approach relies on solving dual degenerate linear programming problems having infinitely many optimal solutions, each of them delivering an associated joint decreasing direction. When the considered MOP is constrained, no additional difficulties appear while using the algorithms we propose (as the feasibility is guaranteed during the procedure of selecting joint decreasing directions without additional costs), while their counterparts from the literature require (usually cost-intensive from a computational point of view) projections on the feasible sets, solving constrained minimax quadratic optimization problems, and/or additional strong hypotheses (such as the compactness of the feasible set). Last but not least, the previously known algorithms for solving MOPs require high precision solvers, especially when the current iterate lies on the boundary (or near the boundary) of the feasible set in order not to lose the feasibility. Our approach does not rely on the necessity of very accurate computational precision due to the fact that a joint decreasing direction is chosen from a cone (instead of being determined by the unique optimal solution of a more complex optimization problem as it is the case for its counterparts from the literature), making thus our approach more robust to external interferences. Numerical experiments on convex unconstrained and convex linearly constrained test MOPs as well on portfolio optimization problems including real stock market data show the applicability and the efficiency (see Tables 1, 2, 3) of the proposed algorithms.

As further research we plan to consider other choices of stepsizes in the proposed iterative methods and to introduce algorithms where quadratic programming problems are used for determining the joint decreasing directions (following Schäffler et al., 2002). In Fliege et al. (2019) an interesting complexity analysis for the steepest descent Algorithm 3.2 is provided for unconstrained MOPs. An investigation of the complexity of the algorithms proposed in this paper is planned as a future research. We also plan to study accelerations of the proposed algorithms (e.g. exploiting Remark 5). It would be also interesting to give a subdivision framework (like in Illés and Lovics (2018)) for all these types of MOPs in order to efficiently approximate their entire weak Pareto frontiers.

CRediT authorship contribution statement

Sorin-Mihai Grad: Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Tibor Illés:** Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization. **Petra Renáta Rigó:** Writing – review & editing, Writing – original draft, Visualization, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Data curation, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was supported by the National Research, Development and Innovation Office (NKFIH) under grant number 2024-1.2.3-HU-RIZONT-2024-00030. The research of Sorin-Mihai Grad was partially supported by a CIAS Senior Research Fellow Grant of the Corvinus Institute for Advanced Studies, and by a public grant from the Fondation Mathématique Jacques Hadamard as part of the Investissement d'avenir project, reference ANR-11-LABX-0056-LMH, LabEx LMH. The research of Tibor Illés was partially supported by the Austrian-Hungarian Action Foundation, project number 116öu8. Petra Renáta Rigó was supported by the János Bolyai Research Scholarship of the Hungarian Academy of Sciences. The comments and suggestions of the handling Associate Editor, of the Co-ordinating Editor in Chief and of five anonymous reviewers are gratefully acknowledged as well, as they aided us in improving the presentation and the integration of our work in the existing body of literature.

References

Ansary, M. A. T., & Panda, G. (2015). A modified Quasi-Newton method for vector optimization problem. *Optimization*, *64*(11), 2289–2306.

Assunção, P. B., Ferreira, O. P., & Prudente, L. F. (2021). Conditional gradient method for multiobjective optimization. *Computational Optimization and Applications*, *78*, 741–768.

Bello Cruz, J. Y., Lucambio Pérez, L. R., & Melo, J. G. (2011). Convergence of the projected gradient method for quasiconvex multiobjective optimization. *Nonlinear Analysis, Theory, Methods and Applications, Series A Theory and Methods*, *74*(16), 5268–5273.

Bonnel, H., Iusem, A. N., & Svaiter, B. F. (2005). Proximal methods in vector optimization. *SIAM Journal on Optimization*, *15*(4), 953–970.

Boţ, R. I., & Grad, S.-M. (2018). Inertial forward–backward methods for solving vector optimization problems. *Optimization*, *67*(7), 959–974.

Boţ, R. I., Grad, S.-M., & Wanka, G. (2009). *Duality in vector optimization*. Berlin-Heidelberg: Springer-Verlag.

Carrizosa, E., & Frenk, J. B. G. (1998). Dominating sets for convex functions with some applications. *Journal of Optimization Theory and Applications*, *96*(2), 281–295.

Chen, J., Tang, L., & Yang, X. (2023). A Barzilai–Borwein descent method for multiobjective optimization problems. *European Journal of Operational Research*, *311*(1), 196–209.

Chen, W., Yang, X. M., & Zhao, Y. (2023). Conditional gradient method for vector optimization. *Computational Optimization and Applications*, *85*(3), 857–896.

Eichfelder, G. (2008). *Adaptive scalarization methods in multiobjective optimization*. Berlin-Heidelberg: Springer-Verlag.

Eschenauer, H., Koski, J., & Osyczka, A. (1990). *Multicriteria design optimization: Procedures and applications*. Berlin: Springer-Verlag.

Evans, G. W. (1984). An overview of techniques for solving multiobjective mathematical programs. *Management Science*, *30*(11), 1268–1282.

(2025). *FICO xpress optimization (version 9.6)*. FICO.

Fliege, J., Graña-Drummond, L. M., & Svaiter, B. F. (2009). Newton's method for multiobjective optimization. *SIAM Journal on Optimization*, *20*(2), 602–626.

Fliege, J., & Svaiter, B. F. (2000). Steepest descent methods for multicriteria optimization. *Mathematical Methods of Operations Research*, *51*(3), 479–494.

Fliege, J., Vaz, A. I. F., & Vicente, L. N. (2019). Complexity of gradient descent for multiobjective optimization. *Optimization Methods & Software*, *34*(5), 949–959.

Fukuda, E. H., & Graña-Drummond, L. M. (2011). On the convergence of the projected gradient method for vector optimization. *Optimization*, *60*(8–9), 1009–1021.

Fukuda, E. H., & Graña-Drummond, L. M. (2013). Inexact projected gradient method for vector optimization. *Computational Optimization and Applications*, *54*(3), 473–493.

Fukuda, E. H., & Graña-Drummond, L. M. (2014). A survey on multiobjective descent methods. *Pesquisa Operacional*, *34*(3), 585–620.

Glineur, F. (2001). *Topics in convex optimization: Interior-point methods, conic duality and approximations* (Ph.D. thesis), Mons, Belgium: University of Mons.

Grad, S.-M. (2019). A survey on proximal point type algorithms for solving vector optimization problems. In *Splitting algorithms, modern operator theory, and applications* (pp. 269–308). Cham: Springer-Verlag.

Graña-Drummond, L. M., & Iusem, A. N. (2004). A projected gradient method for vector optimization problems. *Computational Optimization and Applications*, *28*(1), 5–29.

Graña-Drummond, L. M., Raupp, F. M. P., & Svaiter, B. F. (2014). A quadratically convergent Newton method for vector optimization. *Optimization*, *63*(5), 661–677.

Graña-Drummond, L. M., & Svaiter, B. F. (2005). A steepest descent method for vector optimization. *Journal of Computational and Applied Mathematics*, *175*(2), 395–414.

Illés, T., & Lovics, G. (2018). Approximation of the whole Pareto optimal set for the vector optimization problem. *Acta Polytechnica Hungarica*, *15*(1), 127–148.

Kuhn, H. W., & Tucker, A. W. (1951). Nonlinear programming. In *Proceedings of the 2nd Berkeley symposium on mathematical and statistical probability* (pp. 481–492). University of California Press.

Luc, D. T. (1989). *Lect notes econ math syst: vol. 319, Theory of vector optimization*. Berlin: Springer-Verlag.

Lucambio Pérez, L. R., & Prudente, L. F. (2018). Nonlinear conjugate gradient methods for vector optimization. *SIAM Journal on Optimization*, *28*(3), 2690–2720.

Markowitz, H. M. (1952). Portfolio selection. *The Journal of Finance*, *7*(1), 77–91.

Markowitz, H. M. (1959). *Portfolio selection: Efficient diversification of investment*. New York: John Wiley and Son Inc.

Maros, I. (2003). *Computational techniques of the simplex method*. Dordrecht: Kluwer Academic Publishers.

Miettinen, K. M. (1999). *Nonlinear multiobjective optimization*. Norwell: Kluwer Academic Publishers.

Mukai, H. (1980). Algorithms for multicriterion optimization. *IEEE Transactions on Automatic Control*, *25*, 177–186.

Murty, K. G. (1976). *Linear and combinatorial programming*. New York: John Wiley & Sons, Inc.

Nesterov, Yu (2018). *Springer optim. appl., Lectures on convex optimization*. Switzerland: Springer.

Roos, C., Terlaky, T., & Vial, J.-P. (2005). *Theory and algorithms for linear optimization*. New York: Springer.

Schäffler, S., Schultz, R., & Weinzierl, K. (2002). Stochastic method for the solution of unconstrained vector optimization problems. *Journal of Optimization Theory and Applications*, *114*(1), 209–222.

Vörös, J., Kehl, D., & Rappai, G. (2026). Analytical forms of efficient frontier of euro stoxx 50 and bux 5 portfolios [in hungarian]. *Szigma*, *LVII*(1–2), 115–145.

Vörös, J., & Rappai, G. (2025). The efficient frontiers of the Euro Stoxx 50, the DAX 40 and the BUX portfolios with and without short selling. submitted for publication.

Wu, W., & Yang, X. (2025). A branch and bound algorithm for continuous multiobjective optimization problems using general ordering cones. *European Journal of Operational Research*, *326*(1), 28–41.