A proximal gradient method with an explicit line search for multiobjective optimization

Y. Bello-Cruz[1], J. G. Melo[2], L. F. Prudente[2], R. V. G. Serra[3*]

[1]Department of Mathematical Sciences, Northern Illinois University, Watson Hall 366, DeKalb, 74001-970, IL, USA.
[2]Institute of Mathematics and Statistics, Federal University of Goias, R. Jacarandá , Goiânia, 74001-970, GO, Brazil.
[3*]Department of Mathematics, Federal University of Piauí, R. Dirce Oliveira, 1521-1655, Teresina, 64049550, PI, Brazil.

*Corresponding author(s). E-mail(s): rayserra@ufpi.edu.br;
Contributing authors: yunierbello@niu.edu; jefferson@ufg.br; lfprudente@ufg.br;

Abstract

We present a proximal gradient method for solving convex multiobjective optimization problems, where each objective function is the sum of two convex functions, one of which is assumed to be continuously differentiable. The algorithm incorporates a backtracking line search procedure that requires solving only one proximal subproblem per iteration, and is exclusively applied to the differentiable part of the objective functions. Under mild assumptions, we show that the sequence generated by the method converges to a weakly Pareto optimal point of the problem. Additionally, we establish an iteration complexity bound by proving that the method finds an ε-approximate weakly Pareto point in at most $\mathcal{O}(1/\varepsilon)$ iterations. Numerical experiments illustrating the practical behavior of the method are presented.

Keywords: Convex programming, Full convergence, Proximal Gradient method, Multiobjective optimization.

1 Introduction

Multiobjective optimization involves simultaneously minimizing two or more objective functions. Consider the vector-valued function $F : \mathbb{R}^n \to (\mathbb{R} \cup \{+\infty\})^m$ defined by $F(x) := (F_1(x), \ldots, F_m(x))$. The associated unconstrained multiobjective optimization problem is denoted as

$$\min_{x \in \mathbb{R}^n} F(x). \tag{1}$$

This paper focuses on problem (1), assuming that each component $F_j : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ has the following special separable structure:

$$F_j(x) := G_j(x) + H_j(x), \quad j = 1, \ldots, m, \tag{2}$$

where $G_j : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable and convex, and $H_j : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ is proper, convex, and continuous on its domain, but possibly nonsmooth. This particular problem has many important applications. In particular, when H_j is an indicator function of a convex set $\mathcal{C}$, (1) reduces to the convex-constrained problem of minimizing $G(x) := (G_1(x), \ldots, G_m(x))$ subject to $x \in \mathcal{C}$. Furthermore, the separable structure (2) can be used to model robust multiobjective optimization problems. These problems involve dealing with uncertainty in the data, and the optimal solution must account for the worst possible scenario. We refer the reader to [60] for more details about applications.

Related works. Proximal gradient methods for solving problem (1) with a separable structure as in (2) have been extensively studied in the scalar setting (i.e., $m = 1$); see, for example, [6] and references therein. However, their extension to the multiobjective/vector setting has only recently attracted attention. Specifically, in [18], a forward-backward algorithm based on an a priori linear scalarization approach was proposed and analyzed for convex vector optimization problems in infinite-dimensional spaces. Subsequently, [60] introduced a multiobjective proximal gradient method with an Armijo-type line search and analyzed its convergence without assuming convexity of the objective functions. They demonstrated that every accumulation point of the generated sequence is a weak Pareto critical point. It is worth mentioning that, when the objective function in (1)-(2) contains only a differentiable component (i.e., $H_j = 0$ for all $j = 1, \ldots, m$), this algorithm recovers the multiobjective steepest descent method of [32]. A variant of the proximal gradient algorithm in [60] was proposed and analyzed in [10] for solving convex vector optimization problems. The authors established asymptotic convergence and provided an iteration-complexity bounds for obtaining an approximate stationary solution in the convex case. A key distinction between these algorithms is their line search strategy: the algorithm proposed in [60] employs an Armijo-type line search for the objective function F, while the latter uses a line search based solely on the differentiable component G, extending Beck and Teboulle's backtracking line search [6] to the vector-valued setting. However, the latter algorithm shares the potential drawback of requiring to solve multiple proximal subproblem per iteration, as a proximal step must be computed whenever the line search procedure fails. In [63], the authors provided a convergence rate analysis for the algorithm in [60], establishing sublinear rates in the convex setting and linear rates in the strongly convex case. A multiobjective Barzilai-Borwein-type proximal gradient method was developed in [23], where sublinear and linear convergence rates were established in the nonconvex, convex, and strongly convex settings.

Recently, accelerated first-order methods have been proposed for solving composite multiobjective optimization problems, such as those defined by (1)-(2). These methods enhance the $\mathcal{O}(1/k)$ convergence rate of non-accelerated approaches, achieving a unified global convergence rate of $\mathcal{O}(1/k^2)$. Specifically, [62] introduced an accelerated proximal gradient method, establishing global convergence in terms of a merit function that, in the scalar case, precisely recovers the optimal value gap $F(x^k) - \inf F(x)$. Subsequently, [65] demonstrated that this method can achieve an improved asymptotic convergence rate of $o(1/k^2)$. Other multiobjective accelerated first-order algorithms have been studied in [50, 61]. To address the limitation of the accelerated proximal gradient method in [62], namely that its full convergence remains unproven, [61] proposed a modified version incorporating additional hyperparameters, which ensures convergence under mild assumptions. Furthermore, [50] introduced a multiobjective variant of FISTA that enforces monotonicity in the objective function values while retaining the global convergence rate. Most recently, [66] proposed a multiobjective proximal gradient method and an accelerated variant for solving convex problems as (1)-(2). Both schemes employ an implicit line search rule, avoiding the global Lipschitz assumption on ∇G_j, and achieve sublinear convergence rates. For the unaccelerated version, the authors proved a linear convergence rate under a strong convexity assumption on the smooth components G_j. In addition to these first-order methods, other approaches that utilize either first or second order information about the objective function have been explored for composite multiobjective optimization problems like (1)-(2). These include the generalized conditional gradient method proposed in [5] and subsequently refined by an adaptive variant in [37]. Each iteration of these algorithms solves a linear minimization oracle to generate a search direction, a design that can be particularly advantageous for large scale instances. Armijo-type backtracking strategies are applied to the full objective function. Under the assumptions of convexity, Lipschitz continuity of ∇G_j, and compactness of $\operatorname{dom}(H)$, both variants attain $\mathcal{O}(1/k)$ sublinear convergence rate. Second-order approaches include Newton-type proximal methods [3, 22], and quasi-Newton proximal schemes [44, 51, 52]. Finally, extensions of the multiobjective proximal gradient method of [60] that employ Bregman distances instead of Euclidean norms were studied in [1, 20].

Main contributions. In the present paper, inspired by the work [8], we go a step further than [10] and introduce a proximal gradient method with an explicit line search procedure that is characterized by the following features: (i) only one proximal subproblem is solved per iteration; (ii) the backtracking scheme is exclusively applied to the differentiable component function G. By applying the line search solely to G_j, we effectively leverage first-order information and avoid performing multiple evaluations of potentially costly functions H_j, thereby reducing the per-iteration cost. The line search procedure is essential whenever the global L_j-Lipschitz continuity of the gradients of G_j fails, or even when computing an acceptable upper bound for L_j is challenging. Moreover, even when the Lipschitz constant is known, the line search may allow longer steps toward the solution by using the information at every iteration. To illustrate the practicality and efficiency of the proposed method, in Section 6, we compare it with some multiobjective proximal gradient methods. In this section, we also discuss that when the structure (2) is used to model robust multiobjective optimization problems, the evaluation of H_j often involves solving an auxiliary optimization problem, making our proposal particularly well-suited and efficient in such scenarios. Regarding the convergence analysis of the method, we establish, under mild assumptions, the full convergence of the generated sequence to a weakly Pareto optimal solution of the problem. Additionally, we derive iteration-complexity bounds and show that the method finds an *ε-approximate weakly Pareto point* in at most $\mathcal{O}(1/\varepsilon)$ iterations.

Brief review of classical multiobjective optimization methods. A widely used strategy for solving multiobjective optimization problems has been to extend methods developed for scalar-valued optimization to vector-valued optimization. This approach was introduced in the work of Fliege and Svaiter [32], where a multiobjective steepest descent method was proposed and analyzed. The algorithm was later extended to general vector optimization problems in [41] and adapted for constrained vector optimization problems in [40]. Further studies refined the convergence analysis of this algorithm under quasi-convexity assumptions on the objective function and examined inexact versions of the method [9, 35, 36]. The iteration complexity of the multiobjective steepest descent method was analyzed in [34]. An interior Bregman gradient method was proposed and analyzed in [21] for solving vector optimization problems. The proximal point method for solving vector optimization problems in Hilbert spaces was introduced and analyzed in [17] and further developed in [19, 24, 26]. The convergence of a scalarized version of the multiobjective proximal point method for minimizing quasi-convex and pseudo-convex functions was studied in [13]. The Newton method was extended in [33] to solve multiobjective optimization problems and further generalized for the vector optimization setting in [42]. A unified analysis of the multiobjective Newton method was presented in [64], while a globally convergent Newton-type method was proposed in [39]. Additionally, several variants of quasi-Newton methods have been explored, including those discussed in [53, 55, 56]. Subsequently, [7] investigated the convergence of a subgradient method for vector optimization problems. It was established that every accumulation point of the generated sequence is a weakly efficient optimal point. However, the question of whether the proposed scheme achieves full convergence was not answered. A multiobjective subgradient method based on a variable scalarization approach was introduced in [27] to address quasiconvex nonsmooth multiobjective optimization problems. The authors showed that the sequence generated by this method converges to a Pareto optimal point. In this approach, the search direction is determined dynamically, similar to the one in [40], without relying on fixed linear scalarization. Extensions of the gradient, subgradient, and proximal point methods have been studied in the context of multiobjective and vector optimization problems on Riemannian manifolds; see, for example, [11, 12, 14]. In this framework, the full convergence of the sequences generated by these methods depends significantly on the sectional curvature of the manifold. Specifically, for convex problems, the gradient and subgradient methods require non-negative sectional curvature for convergence, whereas the proximal point method relies on non-positive sectional curvature (i.e., Hadamard manifolds). The classical conjugate gradient method was first analyzed in the context of vector optimization in [48] and further developed in [38]. In [4], the conditional gradient method was introduced for multiobjective optimization and asymptotic convergence and iteration-complexity bounds for computing approximate (stationary) solutions were established. An interior point method was analyzed in [31] in the multiobjective optimization setting.

Organization of the paper. Section 2 presents some definitions and basic results used throughout the paper. In Section 3, we introduce our proximal gradient algorithm with the new explicit line search procedure and show that it is well-defined. The asymptotic convergence analysis is presented in Section 4, and iteration-complexity bounds are established in Section 5. Numerical experiments are presented in Section 6. Finally, Section 7 contains a conclusion.

2 Preliminaries

We write $p := q$ to indicate that p is defined to be equal to q. We denote by $\mathbb{N}$ the set of nonnegative integers $\{0, 1, 2, \ldots\}$, by $\mathbb{R}$ the set of real numbers, and by $\mathbb{R}_+$ the set of nonnegative real numbers. As usual, $\mathbb{R}^m$ and $\mathbb{R}^{m\times n}$ stand for the set of m dimensional real column vectors and the set of $m \times n$ real matrices, respectively. We define $\overline{\mathbb{R}} := \mathbb{R} \cup \{+\infty\}$ and $\overline{\mathbb{R}}^m := (\mathbb{R} \cup \{+\infty\})^m$. For $u, v \in \mathbb{R}^m$, $v \succeq u$ (or $u \preceq v$) means that $v_j \geq u_j$ for $j = 1, \ldots, m$, and $v \succ u$ (or $u \prec v$) means that $v_j > u_j$ for $j = 1, \ldots, m$. The transpose of the vector $u \in \mathbb{R}^m$ is denoted by $u^\top$. The Euclidean norm is denoted by $\|\cdot\|$.

Next we present some definitions and properties for scalar functions. The effective domain of $\phi : \mathbb{R}^n \to \overline{\mathbb{R}}$ is defined as $\operatorname{dom}(\phi) := \{x \in \mathbb{R}^n \mid \phi(x) < +\infty\}$. The function ϕ is said to be *proper* if $\operatorname{dom}(\phi)$ is nonempty, and ϕ is *convex* if, for every $x, y \in \operatorname{dom}(\phi)$ and $t \in [0, 1]$, there holds

$$\phi(tx + (1-t)y) \leq t\phi(x) + (1-t)\phi(y).$$

The subdifferential of a proper convex function ϕ at $x \in \operatorname{dom}(\phi)$ is defined by

$$\partial\phi(x) := \{u \in \mathbb{R}^n \mid \phi(y) \geq \phi(x) + u^\top(y-x), \text{ for all } y \in \mathbb{R}^n\}. \tag{3}$$

If $\phi : \mathbb{R}^n \to \mathbb{R}$ is convex and differentiable, then the subdifferential is an unitary set, i.e., $\partial\phi(x) = \{\nabla\phi(x)\}$, and so

$$\phi(y) \geq \phi(x) + \nabla\phi(x)^\top(y-x), \quad \text{for all } x, y \in \mathbb{R}^n, \tag{4}$$

where $\nabla\phi$ denotes the gradient of ϕ. We say that $\nabla\phi$ is *Lipschitz continuous* with constant $L > 0$ if

$$\|\nabla\phi(y) - \nabla\phi(x)\| \le L\|y - x\|, \quad \text{for all}\, x, y \in \mathbb{R}^n.$$

In this case, it holds that

$$\phi(y) \le \phi(x) + \nabla\phi(x)^\top(y - x) + \frac{L}{2}\|y - x\|^2, \quad \text{for all}\, x, y \in \mathbb{R}^n, \tag{5}$$

see, for example, [15, Proposition A.24].

Let $F : \mathbb{R}^n \to \overline{\mathbb{R}}^m$ be a vector-valued function given by $F(x) := (F_1(x), \ldots, F_m(x))$. The effective domain and the image of F are denoted by $\mathrm{dom}(F) := \{x \in \mathbb{R}^n \mid F_j(x) < +\infty, j = 1, \ldots, m\}$ and $\mathrm{Im}(F) := \{y \in \mathbb{R}^m \mid y = F(x), x \in \mathbb{R}^n\}$, respectively. Since for a multiobjective optimization problem, there is typically no single point minimizing all functions at once, we employ the concept of *Pareto optimality* to characterize a solution, as defined below.

Definition 1. *A point $\bar{x} \in \mathbb{R}^n$ is Pareto optimal for (1) if there does not exist another $x \in \mathbb{R}^n$ such that $F(x) \preceq F(\bar{x})$ and $F_i(x) < F_i(\bar{x})$ for at least one index $i \in \{1, \ldots, m\}$. Furthermore, $\bar{x} \in \mathbb{R}^n$ is said to be a weakly Pareto optimal point for (1) if there does not exist another $x \in \mathbb{R}^n$ such that $F(x) \prec F(\bar{x})$.*

We conclude this section by recalling the well-known concept of quasi-Fejér convergence, see [2, 30].

Definition 2. *Let Ω be a nonempty subset of $\mathbb{R}^n$. A sequence $\{x^k\}$ in $\mathbb{R}^n$ is said to be quasi-Fejér convergent to Ω if and only if for every $x \in \Omega$ there exists a summable sequence $\{\varepsilon_k\} \subset \mathbb{R}_+$ such that, for every $k \in \mathbb{N}$, there holds*

$$\|x^{k+1} - x\|^2 \le \|x^k - x\|^2 + \varepsilon_k.$$

The main property of a quasi-Fejér convergent sequence is stated in the following lemma, and its proof can be found in [57].

Lemma 1. *If $\{x^k\}$ is quasi-Féjer convergent to Ω, then the following statements hold:*
(i) *the sequence $\{x^k\}$ is bounded;*
(ii) *if a limit point x^* of $\{x^k\}$ belongs to Ω, then $\{x^k\}$ converges to x^*.*

3 Algorithm description and well-definedness

For clarity and organization, we explicitly state the assumptions regarding problem (1), which will be considered throughout the article.

General Assumption: *The objective function $F : \mathbb{R}^n \to \overline{\mathbb{R}}^m$, given by $F(x) := (F_1(x), \ldots, F_m(x))$, has the following special separable structure:*

$$F_j(x) := G_j(x) + H_j(x), \quad j = 1, \ldots, m,$$

where:

(i) *$G_j : \mathbb{R}^n \to \mathbb{R}$ is continuously differentiable and convex, for $j = 1, \ldots, m$;*
(ii) *$H_j : \mathbb{R}^n \to \overline{\mathbb{R}}$ is proper, convex, and continuous on $\mathrm{dom}(H_j)$, for $j = 1, \ldots, m$;*
(iii) *$\mathrm{dom}(F)$ is nonempty and closed.*

We now introduce a proximal regularization that will be employed in our method. Given $x \in \mathrm{dom}(F)$ and $\alpha > 0$, let us define the function $\psi_x : \mathbb{R}^n \to \overline{\mathbb{R}}$ as follows:

$$\psi_x(u) := \max_{j=1,\ldots,m} \left(\nabla G_j(x)^\top (u - x) + H_j(u) - H_j(x)\right), \quad \text{for all } u \in \mathbb{R}^n. \tag{6}$$

Now, consider the scalar-valued optimization problem:

$$\min_{u \in \mathbb{R}^n} \psi_x(u) + \frac{1}{2\alpha}\|u - x\|^2. \tag{7}$$

Due to the convexity of H_j for $j = 1, \ldots, m$, it follows that ψ_x is convex and the objective function of (7) is strongly convex. Hence, (7) has a unique solution, which trivially belongs to $\mathrm{dom}(F)$ (noting that $\mathrm{dom}(F) = \mathrm{dom}(\psi_x)$). We denote the solution of (7) by $p_\alpha(x)$ and its optimal value by $\theta_\alpha(x)$, i.e.,

$$p_\alpha(x) := \arg\min_{u \in \mathbb{R}^n} \psi_x(u) + \frac{1}{2\alpha}\|u - x\|^2 \tag{8}$$

and

$$\theta_\alpha(x) := \psi_x(p_\alpha(x)) + \frac{1}{2\alpha}\|p_\alpha(x) - x\|^2. \tag{9}$$

It is worth noting that when $m = 1$, $p_\alpha(x)$ is directly related to the well-known *forward-backward* operator evaluated at x.

In the following, we state some properties concerning the functions $p_\alpha(\cdot)$ and $\theta_\alpha(\cdot)$, defined in (8) and (9), respectively.

Lemma 2. *Given $\alpha > 0$, consider $p_\alpha : \mathrm{dom}(F) \to \mathrm{dom}(F)$ and $\theta_\alpha : \mathrm{dom}(F) \to \mathbb{R}$ as in (8) and (9), respectively. Then, the following statements are true.*
(i) *$\theta_\alpha(x) \le 0$ for all $x \in \mathrm{dom}(F)$.*
(ii) *The following statements are equivalent: (a) x is a weakly Pareto optimal point of (1); (b) $\theta_\alpha(x) = 0$; (c) $p_\alpha(x) = x$.*
(iii) *$p_\alpha(\cdot)$ and $\theta_\alpha(\cdot)$ are continuous.*

Proof. See [60, Lemma 3.2]. □

3.1 Algorithm

We now formally describe our proximal gradient method for solving (1).

MPG-Explicit: Multiobjective Proximal Gradient algorithm with Explicit Line search

---

Step 0. Let $x^0 \in \mathrm{dom}(F)$, $\alpha > 0$, $\gamma \in (0, 2/\alpha)$, and $0 < \tau_1 < \tau_2 < 1$ be given. Initialize $k \leftarrow 0$.

Step 1. *Subproblem*
Compute $p^k := p_\alpha(x^k)$ and $\theta_\alpha(x^k)$ as in (8) and (9), respectively.
Step 2. *Stopping criterion*
If $\theta_\alpha(x^k) = 0$, then STOP.
Step 3. *Line search procedure*
Define $d^k := p^k - x^k$, take $j_k^* \in \operatorname{argmax}_{j=1,\ldots,m} \nabla G_j(x^k)^\top d^k$, and set $t_{\text{trial}} = 1$.

Step 3.1. If

$$G_{j_k^*}(x^k + t_{\text{trial}} d^k) \leq G_{j_k^*}(x^k) + t_{\text{trial}} \nabla G_{j_k^*}(x^k)^\top d^k + t_{\text{trial}} \frac{\gamma}{2}\|d^k\|^2,$$

then go to Step 3.2. Otherwise, compute $t_{\text{new}} \in [\tau_1 t_{\text{trial}}, \tau_2 t_{\text{trial}}]$, set $t_{\text{trial}} \leftarrow t_{\text{new}}$ and repeat Step 3.1.
Step 3.2. If

$$F(x^k + t_{\text{trial}} d^k) \preceq F(x^k),$$

then define $t_k = t_{\text{trial}}$ and go to Step 4.
Step 3.3. Compute $t_{\text{new}} \in [\tau_1 t_{\text{trial}}, \tau_2 t_{\text{trial}}]$ and set $t_{\text{trial}} \leftarrow t_{\text{new}}$. If

$$G_j(x^k + t_{\text{trial}} d^k) \leq G_j(x^k) + t_{\text{trial}} \nabla G_j(x^k)^\top d^k + t_{\text{trial}} \frac{\gamma}{2}\|d^k\|^2, \quad j = 1, \ldots, m,$$

then define $t_k = t_{\text{trial}}$ and go to Step 4. Otherwise, repeat Step 3.3.

Step 4. *Iterate*
Define $x^{k+1} := x^k + t_k d^k$, set $k \leftarrow k + 1$ and go to Step 1.

Some comments are in order. (a) If H_j lacks smoothness, the subproblem (7) in Step 1 may also lack smoothness. In such instances, the approach for solving (7) depends on the specific structure of the functions H_j. In Section 6, we will explore an application to robust multiobjective optimization and discuss how to solve the subproblem in this particular scenario. (b) At Step 2, it follows from Lemma 2 (ii) that the MPG-Explicit algorithm stops at iteration k if and only if x^k is a weakly Pareto optimal point. (c) In the line search procedure, the backtracking scheme is applied only to the differentiable functions G_j, preventing the computation of potentially costly H_j functions. In Step 3.1, we first perform a line search using a single function $G_{j_k^*}$. If the resulting t_{trial} stepsize leads to a decrease in all F_j's, it is accepted and the line search is finished. Otherwise, the line search is carried out on all G_j's functions until the condition in Step 3.3 is satisfied. Thus, the k-th iteration concludes with one of the following scenarios:

$$G_{j_k^*}(x^{k+1}) \leq G_{j_k^*}(x^k) + t_k \nabla G_{j_k^*}(x^k)^\top d^k + t_k \frac{\gamma}{2}\|d^k\|^2 \text{ and } \left[F(x^{k+1}) \preceq F(x^k)\right], \quad (10)$$

or

$$G_j(x^{k+1}) \leq G_j(x^k) + t_k \nabla G_j(x^k)^\top d^k + t_k \frac{\gamma}{2}\|d^k\|^2, \quad j = 1, \ldots, m. \quad (11)$$

It is worth mentioning that, given $j \in \{1, \ldots, m\}$, d^k is not necessarily a descent direction for G_j at x^k, meaning that it can happen that $\nabla G_j(x^k)^\top d^k \geq 0$. However, as we will see, both cases (10) and (11) result in a decrease in the value of each objective F_j.

3.2 Well-definedness analysis

In the following, we show that the MPG-Explicit algorithm is well defined. This means that if the algorithm does not stop at x^k, then it is possible to obtain x^{k+1} in a finite time.

Theorem 3. *The MPG-Explicit algorithm is well defined and stops at iteration k if and only if x^k is a weakly Pareto optimal point.*

Proof. Due to the strong convexity of the objective function in (7), the subproblem at Step 1 is solvable, allowing the computation of p^k and $\theta_\alpha(x^k)$. According to Lemma 2 (ii), the MPG-Explicit algorithm stops at Step 2 in iteration k if and only if x^k is a weakly Pareto optimal point. Now, assuming that x^k is not a weakly Pareto optimal point, it follows from Lemma 2 (ii) that $d^k \neq 0$. Thus, for any arbitrary index $j \in \{1, \dots, m\}$, the differentiability of G_j ensures the existence of $\delta > 0$ such that

$$\left| \frac{G_j(x^k + td^k) - G_j(x^k)}{t} - \nabla G_j(x^k)^\top d^k \right| \leq \frac{\gamma}{2}\|d^k\|^2,$$

for all $t \in (0, \delta]$. Hence,

$$G_j(x^k + td^k) \leq G_j(x^k) + t\nabla G_j(x^k)^\top d^k + t\frac{\gamma}{2}\|d^k\|^2, \quad j = 1, \dots, m,$$

for all $t \in (0, \delta]$. Therefore, the line search procedure ultimately finishes in a finite number of (inner) steps, and x^{k+1} is properly defined at Step 4. □

4 Asymptotic convergence analysis

The MPG-Explicit algorithm successfully stops if a weakly Pareto optimal point is found. Then, in order to analyze its convergence properties, we assume henceforth that the MPG-Explicit algorithm generates an infinite sequence, which is equivalent to saying that no x^k is a weakly Pareto optimal point of problem (1).

In the following, we establish some key inequalities for our analysis. In particular, we show that if, at iteration k, the backtracking in the line search procedure is based on G_j, it automatically leads to a decrease in the corresponding objective F_j.

Lemma 4. *Let $\{x^k\}$ be generated by the MPG-Explicit algorithm. Suppose that at iteration k, the index $j \in \{1, \dots, m\}$ is such that*

$$G_j(x^{k+1}) \leq G_j(x^k) + t_k\nabla G_j(x^k)^\top d^k + t_k\frac{\gamma}{2}\|d^k\|^2. \tag{12}$$

Then

(i) $\psi_{x^k}(p^k) \geq \frac{1}{t_k}\left(F_j(x^{k+1}) - F_j(x^k)\right) - \frac{\gamma}{2}\|d^k\|^2$;

(ii) *for every $x \in \mathrm{dom}(F)$, we have*

$$\begin{aligned}\|x^{k+1} - x\|^2 \le \|x^k - x\|^2 &+ 2\alpha \left(F_j(x^k) - F_j(x^{k+1})\right) + 2\alpha t_k \max_{i=1,\ldots,m} \left(F_i(x) - F_i(x^k)\right) \\ &- 2\alpha t_k \left(\frac{1}{\alpha} - \frac{\gamma}{2}\right) \|d^k\|^2 + t_k^2 \|d^k\|^2;\end{aligned} \tag{13}$$

(iii) $F_j(x^{k+1}) - F_j(x^k) \le -t_k \left(\dfrac{2 - \gamma\alpha}{2\alpha}\right) \|d^k\|^2$.

Proof. (i) It follows from (12) that

$$\nabla G_j(x^k)^\top d^k \ge \frac{1}{t_k}\left(G_j(x^{k+1}) - G_j(x^k)\right) - \frac{\gamma}{2}\|d^k\|^2.$$

Hence, in view of the definition of ψ_{x^k} in (6), we have

$$\begin{aligned}\psi_{x^k}(p^k) &\ge \nabla G_j(x^k)^\top d^k + H_j(p^k) - H_j(x^k) \\ &\ge \frac{1}{t_k}\left[G_j(x^{k+1}) - G_j(x^k) + t_k\left(H_j(p^k) - H_j(x^k)\right)\right] - \frac{\gamma}{2}\|d^k\|^2.\end{aligned}$$

Since $x^{k+1} = x^k + t_k(p^k - x^k)$ with $t_k \in (0,1]$, by the convexity of H_j, we have

$$t_k\left(H_j(p^k) - H_j(x^k)\right) \ge H_j(x^{k+1}) - H_j(x^k).$$

By combining the latter two inequalities and using that $F_j = G_j + H_j$, we obtain the desired result.
(ii) Let $x \in \mathrm{dom}(F)$. In view of the definition of x^{k+1} in Step 4, we have

$$\begin{aligned}\|x^{k+1} - x\|^2 &= \|x^k - x\|^2 + \|x^{k+1} - x^k\|^2 + 2(x^{k+1} - x^k)^\top (x^k - x) \\ &= \|x^k - x\|^2 + t_k^2\|d^k\|^2 + 2t_k (d^k)^\top (x^k - x).\end{aligned} \tag{14}$$

Our goal now is to appropriately estimate the quantity $2t_k(d^k)^\top(x^k - x)$. The first-order optimality condition of (8) implies that $-(d^k/\alpha) \in \partial\psi_{x^k}(p^k)$. Hence, by the subgradient inequality (3) and using item (i), we have

$$\begin{aligned}\psi_{x^k}(x) &\ge \psi_{x^k}(p^k) + \frac{1}{\alpha}(d^k)^\top(p^k - x) \\ &= \psi_{x^k}(p^k) + \frac{1}{\alpha}(d^k)^\top(x^k - x) + \frac{1}{\alpha}\|d^k\|^2 \\ &\ge \frac{1}{t_k}\left(F_j(x^{k+1}) - F_j(x^k)\right) + \frac{1}{\alpha}(d^k)^\top(x^k - x) + \left(\frac{1}{\alpha} - \frac{\gamma}{2}\right)\|d^k\|^2.\end{aligned} \tag{15}$$

Now, in view of the definition of ψ_x in (6), the gradient inequality (4) with $\phi = G_i$, and the fact that $F_i = G_i + H_i$, $i = 1, \ldots, m$, we have $\max_{i=1,\ldots,m}\left(F_i(x) - F_i(x^k)\right) \ge \psi_{x^k}(x)$, which combined with (15) imply that

$$2t_k(d^k)^\top(x^k - x) \le 2\alpha\left(F_j(x^k) - F_j(x^{k+1})\right) + 2\alpha t_k \max_{i=1,\ldots,m}\left(F_i(x) - F_i(x^k)\right) \\ - 2t_k\left(1 - \frac{\alpha\gamma}{2}\right)\|d^k\|^2.$$

This inequality together with (14) yields (13).

(iii) By setting $x = x^k$ in (13) and taking into account that $\|x^{k+1} - x^k\|^2 = t_k^2\|d^k\|^2$, some algebraic manipulations give the desired inequality. □

Remark 1. *Given that, at iteration k, the line search procedure terminates with a stepsize t_k satisfying either (10) or (11), the inequality (12) holds for $j = j_k^* \in \operatorname{argmax}_{j=1,\ldots,m} \nabla G_j(x^k)^\top d^k$. Consequently, Lemma 4 is always applicable to the index j_k^*.*

In the following lemma, we show that the objective function sequence $\{F_j(x^k)\}$ is nonincreasing for each $j = 1, \ldots, m$.

Lemma 5. *Let $\{x^k\}$ be generated by the MPG-Explicit algorithm. Then, for $j = 1, \ldots, m$, it holds that $\{F_j(x^k)\}$ is a non-increasing sequence, i.e.,*

$$F(x^{k+1}) \preceq F(x^k), \quad \text{for all } k \in \mathbb{N}.$$

Proof. Let $k \in \mathbb{N}$. We consider two cases: (i) (10) holds at iteration k; (ii) (11) holds at iteration k. In the first case, then statement of the lemma is a straightforward consequence of the second condition in (10). Consider now the second case. Taking into account that $\gamma < 2/\alpha$, it follows from Lemma 4(iii) that $F_j(x^{k+1}) < F_j(x^k)$ for $j = 1, \ldots, m$, concluding the proof. □

To establish the convergence of the MPG-Explicit algorithm, we introduce the following additional assumption, meaning that the set $\operatorname{Im}(F)$ is *complete* with respect to the *Paretian* cone $\mathbb{R}^m_+ := \{z \in \mathbb{R}^m \mid z \succeq 0\}$.

Assumption (A1): *All monotonically nonincreasing sequences in the set Im(F) are bounded below by a point in Im(F), i.e., for every sequence $\{y^k\} \subset \operatorname{dom}(F)$ such that $F(y^{k+1}) \preceq F(y^k)$ for all $k \in \mathbb{N}$, there exists $y \in \operatorname{dom}(F)$ such that $F(y) \preceq F(y^k)$ for all $k \in \mathbb{N}$.*

Assumption (A1) is widely used in the related literature (e.g., [9, 10, 13, 17–19, 24, 25, 36, 40, 41]). It is commonly employed to ensure the existence of efficient solutions for vector-valued optimization problems, see [47, Section 2.3]. In the scalar-valued unconstrained case, it is equivalent to ensuring the existence of an optimal point.

Theorem 6. *Let $\{x^k\}$ be generated by the MPG-Explicit algorithm and suppose that Assumption (A1) holds. Then, $\{x^k\}$ converges to a weakly Pareto optimal point $\bar{x}$ of problem (1).*

Proof. First, we define

$$\Omega := \{x \in \mathrm{dom}(F) \mid F(x) \preceq F(x^k), \text{ for all } k \in \mathbb{N}\}.$$

In view of Lemma 5, it follows from Assumption (A1) that Ω is nonempty. Thus, let us consider an arbitrary $\hat{x} \in \Omega$, i.e., $F(\hat{x}) \preceq F(x^k)$ for all $k \in \mathbb{N}$. Therefore, taking into account Remark 1 and that $\gamma < 2/\alpha$, applying Lemma 4(ii) with $j = j_k^*$ and $x = \hat{x}$ yields

$$\|x^{k+1} - \hat{x}\|^2 \leq \|x^k - \hat{x}\|^2 + \varepsilon_k, \quad \text{for all } k \in \mathbb{N}. \tag{16}$$

where

$$\varepsilon_k := 2\alpha \left(F_{j_k^*}(x^k) - F_{j_k^*}(x^{k+1})\right) + t_k^2\|d^k\|^2, \quad \text{for all } k \in \mathbb{N}. \tag{17}$$

Note that Lemma 5 implies that $\varepsilon_k \geq 0$ for every $k \in \mathbb{N}$. Moreover, since $t_k^2 \leq t_k$, we obtain from Lemma 4(iii) with $j = j_k^*$ that

$$t_k^2 \left(\frac{1}{\alpha} - \frac{\gamma}{2}\right) \|d^k\|^2 \leq t_k \left(\frac{1}{\alpha} - \frac{\gamma}{2}\right) \|d^k\|^2 \leq F_{j_k^*}(x^k) - F_{j_k^*}(x^{k+1}), \quad \text{for all } k \in \mathbb{N},$$

which yields

$$t_k^2\|d^k\|^2 \leq \frac{2\alpha}{2 - \alpha\gamma} \left(F_{j_k^*}(x^k) - F_{j_k^*}(x^{k+1})\right), \quad \text{for all } k \in \mathbb{N}.$$

Hence, by (17) and Lemma 5, we have

$$\varepsilon_k \leq c\left(F_{j_k^*}(x^k) - F_{j_k^*}(x^{k+1})\right) \leq c\sum_{j=1}^{m} \left(F_j(x^k) - F_j(x^{k+1})\right), \quad \text{for all } k \in \mathbb{N},$$

where $c := 2\alpha + 2\alpha/(2 - \alpha\gamma) > 0$. By summing this expression over all indices less than or equal to $N \in \mathbb{N}$, and taking into account that $\hat{x} \in \Omega$, we obtain

$$\begin{aligned}\sum_{k=0}^{N} \varepsilon_k &\leq c\sum_{j=1}^{m}\sum_{k=0}^{N} \left(F_j(x^k) - F_j(x^{k+1})\right) = c\sum_{j=1}^{m} \left(F_j(x^0) - F_j(x^{N+1})\right) \\ &\leq c\sum_{i=j}^{m} \left(F_j(x^0) - F_j(\hat{x})\right) < +\infty.\end{aligned}$$

Therefore, $\sum_{k=0}^{+\infty} \varepsilon_k < +\infty$. Hence, it follows from (16) that $\{x^k\}$ is quasi-Fejér convergent to Ω, see Definition 2. From Lemma 1(i), it follows that $\{x^k\}$ is bounded. Let $\bar{x}$ be a limit point of $\{x^k\}$. Since $\mathrm{dom}(F)$ is closed, F is continuous on $\mathrm{dom}(F)$, and $F(x^{k+1}) \preceq F(x^k)$ for all $k \in \mathbb{N}$, we have $\bar{x} \in \Omega$. Thus, Lemma 1(ii) implies that the whole sequence $\{x^k\}$ converges to $\bar{x}$.

Let us now show that $\bar{x}$ is a weakly Pareto optimal point of problem (1). Since $t_k\|d^k\| = \|x^{k+1} - x^k\|$ and $\{x^k\}$ converges to $\bar{x}$, we obtain

$$\lim_{k \to +\infty} t_k\|d^k\| = 0. \tag{18}$$

We now claim that there exists a subsequence $\{d^{k_\ell}\}$ such that $\lim_{\ell\to+\infty}\|d^{k_\ell}\| = 0$. Indeed, if $\limsup_{k\to+\infty} t_k > 0$, then the claim clearly holds from (18). Hence, assume that $\limsup_{k\to+\infty} t_k = 0$, which in turn implies that $\lim_{k\to+\infty} t_k = 0$. Without loss of generality, suppose that $t_k < 1$ for all $k \in \mathbb{N}$. Therefore, by Step 3 of the algorithm and taking into account Lemma 4(iii), for each $k \in \mathbb{N}$ there exist at least one index $i_k \in \{1, \ldots, m\}$ and

$$0 < \bar{t}_k \leq \frac{t_k}{\tau_1} \tag{19}$$

such that

$$G_{i_k}(x^k + \bar{t}_k d^k) > G_{i_k}(x^k) + \bar{t}_k \nabla G_{i_k}(x^k)^\top d^k + \bar{t}_k \frac{\gamma}{2}\|d^k\|^2.$$

Since $\{1, \ldots, m\}$ is finite, there exist $i_* \in \{1, \ldots, m\}$ and an infinite set of indices $\{k_\ell\} \subset \mathbb{N}$ such that

$$G_{i_*}(x^{k_\ell} + \bar{t}_{k_\ell} d^{k_\ell}) - G_{i_*}(x^{k_\ell}) > \bar{t}_{k_\ell} \nabla G_{i_*}(x^{k_\ell})^\top d^{k_\ell} + \bar{t}_{k_\ell}\frac{\gamma}{2}\|d^{k_\ell}\|^2, \quad \text{for all } \ell \in \mathbb{N}.$$

On the other hand, since G_{i_*} is convex, it follows from (4) that

$$G_{i_*}(x^{k_\ell} + \bar{t}_{k_\ell} d^{k_\ell}) - G_{i_*}(x^{k_\ell}) \leq \bar{t}_{k_\ell} \nabla G_{i_*}(x^{k_\ell} + \bar{t}_{k_\ell} d^{k_\ell})^\top d^{k_\ell}, \quad \text{for all } \ell \in \mathbb{N}.$$

By combining these two inequalities, we obtain

$$\frac{\gamma}{2}\|d^{k_\ell}\|^2 < \left(\nabla G_{i_*}(x^{k_\ell} + \bar{t}_{k_\ell} d^{k_\ell}) - \nabla G_{i_*}(x^{k_\ell})\right)^\top d^{k_\ell}, \quad \text{for all } \ell \in \mathbb{N}.$$

Therefore, by the Cauchy-Schwarz inequality, we have

$$\|d^{k_\ell}\| < \frac{2}{\gamma}\left\|\nabla G_{i_*}(x^{k_\ell} + \bar{t}_{k_\ell} d^{k_\ell}) - \nabla G_{i_*}(x^{k_\ell})\right\|, \quad \text{for all } \ell \in \mathbb{N}.$$

Note that (18) and (19) imply that $\lim_{\ell\to+\infty} \bar{t}_{k_\ell} d^{k_\ell} = 0$. Thus, since G_{i_*} is continuously differentiable, by taking limits as $\ell \to +\infty$ in the latter inequality, we conclude that $\lim_{\ell\to+\infty}\|d^{k_\ell}\| = 0$, as claimed. Hence, it follows from the definition of d^k and Lemma 2(iii) that

$$0 = \lim_{\ell\to+\infty}\|d^{k_\ell}\| = \lim_{\ell\to+\infty}\|p_\alpha(x^{k_\ell}) - x^{k_\ell}\| = \|p_\alpha(\bar{x}) - \bar{x}\|.$$

Therefore, $p_\alpha(\bar{x}) = \bar{x}$, and then, in view of Lemma 2(ii), we conclude that $\bar{x}$ is a weakly Pareto optimal point of problem (1). $\square$

5 Iteration-complexity for MPG-Explicit Algorithm

In this section, we establish some iteration-complexity bounds for the MPG-Explicit algorithm to obtain an approximate weakly Pareto optimal point of (1).

We start by presenting a basic result that will be used to establish the iteration-complexity bounds for the MPG-Explicit algorithm.

Lemma 7. *Assume that (A1) holds and let $\bar{x}$ be the limit point of the sequence $\{x^k\}$ generated by the MPG-Explicit algorithm. Then, for every $N \in \mathbb{N}$, we have*

$$\sum_{k=0}^{N} \|x^{k+1} - x^k\|^2 \leq \frac{2\alpha}{2-\gamma\alpha} \sum_{\ell=1}^{m} \left(F_\ell(x^0) - F_\ell(\bar{x})\right), \tag{20}$$

$$2\alpha \left(\sum_{k=0}^{N} t_k\right) F_{min}^N \leq \bar{d}_0, \tag{21}$$

where

$$F_{min}^N := \min_{j=1,\ldots,m} (F_j(x^N) - F_j(\bar{x})), \quad \bar{d}_0 := \|x^0 - \bar{x}\|^2 + \frac{2\alpha(3-\gamma\alpha)}{2-\gamma\alpha} \sum_{\ell=1}^{m} \left(F_\ell(x^0) - F_\ell(\bar{x})\right).$$

Proof. Let $\bar{x}$ be the limit point of $\{x^k\}$. Since $F(x^k) - F(x^{k+1}) \succeq 0$ for all $k \in \mathbb{N}$, we immediately $F_j(x^k) - F_j(x^{k+1}) \leq \sum_{\ell=1}^{m} (F_\ell(x^k) - F_\ell(x^{k+1}))$, for all $j \in \{1, \ldots, m\}$ and $k \in \mathbb{N}$. Hence, Lemma 4(iii) and the facts that $2 - \gamma\alpha > 0$ and $t_k^2 \leq t_k$ imply that

$$t_k^2 \|d^k\|^2 \leq \frac{2\alpha}{2-\gamma\alpha} \sum_{\ell=1}^{m} \left(F_\ell(x^k) - F_\ell(x^{k+1})\right). \tag{22}$$

Adding both sides of the above inequality from $k = 0, \ldots, N$, we obtain

$$\sum_{k=0}^{N} t_k^2 \|d^k\|^2 \leq \frac{2\alpha}{2-\gamma\alpha} \sum_{\ell=1}^{m} \sum_{k=0}^{N} \left(F_\ell(x^k) - F_\ell(x^{k+1})\right) = \frac{2\alpha}{2-\gamma\alpha} \sum_{\ell=1}^{m} \left(F_\ell(x^0) - F_\ell(x^{N+1})\right).$$

Thus, (20) follows from the above inequality and the facts that $x^{k+1} - x^k = t_k d^k$ and $F(\bar{x}) \preceq F(x^{N+1})$. Now, Lemma 4 (ii) with $x = \bar{x}$ and the fact that $t_k(2 - \gamma\alpha) > 0$, imply that

$$\begin{aligned} \|x^{k+1} - \bar{x}\|^2 \leq & \|x^k - \bar{x}\|^2 + 2\alpha \sum_{\ell=1}^{m} \left(F_\ell(x^k) - F_\ell(x^{k+1})\right) \\ & + 2\alpha t_k \max_{i=1,\ldots,m} \left(F_i(\bar{x}) - F_i(x^k)\right) + t_k^2 \|d^k\|^2, \end{aligned} \tag{23}$$

which in view of (22) and the fact that $F_{min}^k = -\max_{j=1,\ldots m}(F_i(\bar{x}) - F_i(x^k))$, yields

$$2\alpha t_k F_{min}^k \leq \|x^k - \bar{x}\|^2 - \|x^{k+1} - \bar{x}\|^2 + \frac{2\alpha(3-\gamma\alpha)}{2-\gamma\alpha} \sum_{\ell=1}^{m} \left(F_\ell(x^k) - F_\ell(x^{k+1})\right).$$

Adding both sides of the above inequality from $k = 0, \dots, N$, and noting that $F(\bar{x}) \preceq F(x^N) \preceq F(x^k)$ (in particular, $F^N_{min} \leq F^k_{min}$), we obtain

$$2\alpha \left(\sum_{k=0}^{N} t_k \right) F^N_{min} \leq \|x^0 - \bar{x}\|^2 - \|x^{N+1} - \bar{x}\|^2 + \frac{2\alpha(3-\gamma\alpha)}{2-\gamma\alpha} \sum_{\ell=1}^{m} \big(F_\ell(x^0) - F_\ell(x^{N+1}) \big)$$
$$\leq \|x^0 - \bar{x}\|^2 + \frac{2\alpha(3-\gamma\alpha)}{2-\gamma\alpha} \sum_{\ell=1}^{m} \big(F_\ell(x^0) - F_\ell(\bar{x}) \big) = \bar{d}_0,$$

proving that (21) holds. □

Recall that $x^{k+1} = x^k + t_k(p^k - x^k)$, and hence, in view of Lemma 2(ii), the stopping criterion in Step 2 of the MPG-Explicit algorithm is equivalent to $x^{k+1} = x^k$. In our numerical section, in order to compare the MPG-Explicit algorithm with other algorithms, we use the stopping criterion $\|x^{k+1} - x^k\| < \varepsilon$, for a given tolerance $\varepsilon > 0$. In the next result, we establish the iteration-complexity bound for the MPG-Explicit algorithm in order to satisfy this stopping criterion.

Proposition 8. *Assume that (A1) holds and let $\bar{x}$ be the limit point of the sequence $\{x^k\}$ generated by the MPG-Explicit algorithm. Then, for a given tolerance $\varepsilon > 0$, there holds $\|x^{k+1} - x^k\| \leq \varepsilon$ in at most $\mathcal{O}(1/\varepsilon^2)$ iterations.*

Proof. Note that if $\|x^{k+1} - x^k\| > \varepsilon$ for every $k = 0, \dots, N$, then (20) implies that

$$(N+1)\varepsilon^2 < \sum_{k=0}^{N} \|x^{k+1} - x^k\|^2 \leq \frac{2\alpha}{2-\gamma\alpha} \sum_{\ell=1}^{m} \big(F_\ell(x^0) - F_\ell(\bar{x}) \big),$$

which implies that

$$N + 1 < \frac{1}{\varepsilon^2(2-\gamma\alpha)} \sum_{\ell=1}^{m} \big(F_\ell(x^0) - F_\ell(\bar{x}) \big).$$

Hence, if the MPG-Explicit algorithm executes N iterations, where N is the first natural number larger than the right-hand side of the above inequality, we necessarily have $\|x^{k+1} - x^k\| \leq \varepsilon$. Therefore, the proof of the proposition follows. □

Note that (21) can be used to establish a convergence rate on the sequence $\{F^k_{min}\}$ as long as the sequence of stepsizes $\{t_k\}$ stays away from zero. In order to ensure this condition, we need to introduce the following additional assumption.

Assumption (A2): *The gradient ∇G_j is Lipschitz continuous with constant $L_j > 0$, for $j = 1, \dots, m$.*

The next result shows that, under Assumption (A2), the sequence of stepsize $\{t_k\}$ remains bounded away from zero.

Lemma 9. *Let $\{x^k\}$ be generated by the MPG-Explicit algorithm and suppose that Assumption (A2) holds. Then, for all $k \in \mathbb{N}$, we have $t_k \geq t_{\min} := \min\{1, \tau_1\gamma/L_{\max}\}$, where $L_{\max} := \max_{j=1,\dots,m} L_j$.*

Proof. Let $k \in \mathbb{N}$. If $t_k = 1$, then the required inequality trivially holds. Thus, assume that $t_k < 1$. Therefore, by Step 3 of the algorithm, there exist at least one index $i_k \in \{1, \dots, m\}$ and $0 < \bar{t}_k \leq t_k/\tau_1$ such that

$$G_{i_k}(x^k + \bar{t}_k d^k) > G_{i_k}(x^k) + \bar{t}_k \nabla G_{i_k}(x^k)^\top d^k + \bar{t}_k \frac{\gamma}{2} \|d^k\|^2.$$

On the other hand, considering Assumption (A2), it follows from (5) that

$$G_{i_k}(x^k + \bar{t}_k d^k) \leq G_{i_k}(x^k) + \bar{t}_k \nabla G_{i_k}(x^k)^\top d^k + \bar{t}_k^2 \frac{L_{\max}}{2} \|d^k\|^2.$$

By combining the two latter inequalities, we easily obtain that $\bar{t}_k > \gamma/L_{\max}$. Hence, by the definition of $\bar{t}_k$, we have

$$t_k > \frac{\tau_1 \gamma}{L_{\max}},$$

and the proof is concluded. $\square$

The next result presents a sublinear convergence rate on the sequence of minimum optimal values $\{F^k_{min}\}$ defined in Lemma 7. As a consequence, an iteration-complexity bound on this sequence is obtained.

Proposition 10. *Assume that (A1) and (A2) hold and let $\bar{x}$ be the limit point of the sequence $\{x^k\}$ generated by the MPG-Explicit algorithm. Then for every $N \in \mathbb{N}$, we have*

$$F^N_{min} \leq \frac{\bar{d}_0}{2\alpha t_{min}(N+1)},$$

where $t_{\min} = \min\{1, \tau_1\gamma/L_{\max}\}$ and $\bar{d}_0$ is as in (21). As a consequence, given a tolerance $\varepsilon > 0$, we have $F^N_{min} < \varepsilon$ in at most $N = \mathcal{O}(1/\varepsilon)$ iterations.

Proof. The proof of the first statement of the proposition follows immediately from (21) and the fact that Lemma 9 implies that $t_k \geq t_{min}$ for all $k \in \mathbb{N}$. The proof of the second statement follows immediately from the first one. $\square$

Recall that the MPG-Explicit algorithm stops if and only if $\theta_\alpha(x^k) = 0$, which in view of Lemma 2(ii) is equivalent to say that x^k is a weakly Pareto optimal point of (1). In this sense, given a tolerance $\varepsilon > 0$, it seems reasonable to say that a point $\tilde{x}$ satisfying $|\theta_\alpha(\tilde{x})| \leq \varepsilon$ is an $\varepsilon-$approximate weakly Pareto optimal point of (1). In the following, we establish an iteration-complexity bound for the MPG-Explicit algorithm to obtain an $\varepsilon-$approximate weakly Pareto optimal point of (1). We start by proving an auxiliary result which helps to show that $|\theta_\alpha(x^k)|$ is summable.

Lemma 11. *Let $\{x^k\}$ be generated by the MPG-Explicit algorithm and suppose that Assumption (A2) holds. Then, there exists $c > 0$ such that*

$$|\theta_\alpha(x^k)| \leq c \left(F_{j^*_k}(x^k) - F_{j^*_k}(x^{k+1})\right), \quad \text{for all}\, k \in \mathbb{N},$$

where $j^*_k \in \operatorname{argmax}_{j=1,\dots,m} \nabla G_j(x^k)^\top d^k$.

Proof. From the definition of $\theta_\alpha(x^k)$ in (9), Lemma 4(i) with $j = j_k^*$, and taking into account that $\gamma < 2/\alpha$, we obtain

$$\begin{aligned}\theta_\alpha(x^k) &\geq \frac{1}{t_k}\left(F_{j_k^*}(x^{k+1}) - F_{j_k^*}(x^k)\right) + \left(\frac{1}{2\alpha} - \frac{\gamma}{2}\right)\|d^k\|^2\\ &\geq \frac{1}{t_k}\left(F_{j_k^*}(x^{k+1}) - F_{j_k^*}(x^k)\right) - \frac{1}{2\alpha}\|d^k\|^2, \quad \text{for all}\, k \in \mathbb{N}.\end{aligned}$$

Therefore, it follows from Lemma 4(iii) with $j = j_k^*$ that

$$\theta_\alpha(x^k) \geq \left(1 + \frac{1}{2-\gamma\alpha}\right)\frac{1}{t_k}\left(F_{j_k^*}(x^{k+1}) - F_{j_k^*}(x^k)\right), \quad \text{for all}\, k \in \mathbb{N}.$$

Hence, by Lemma 9 and taking into account that $F_{j_k^*}(x^{k+1}) - F_{j_k^*}(x^k) < 0$, we have

$$\theta_\alpha(x^k) \geq \left(1 + \frac{1}{2-\gamma\alpha}\right)\frac{1}{t_{\min}}\left(F_{j_k^*}(x^{k+1}) - F_{j_k^*}(x^k)\right), \quad \text{for all}\, k \in \mathbb{N}.$$

Therefore, we conclude the proof by defining $c := [1 + 1/(2-\gamma\alpha)]/t_{\min}$ and by noting that $c > 0$ and $\theta_\alpha(x^k) < 0$. □

We are now able to establish the iteration-complexity bound for the MPG-Explicit algorithm.

Theorem 12. *Suppose that Assumptions (A1) and (A2) hold. Then, for a given tolerance $\varepsilon > 0$, the MPG-Explicit algorithm generates a point x^k such that $|\theta_\alpha(x^k)| \leq \varepsilon$ in at most $\mathcal{O}(1/\varepsilon)$ iterations.*

Proof. Assume that none of the generated points x^k, $k = 0, \ldots, N$, is an ε-approximate weakly Pareto optimal point of (1), i.e., $|\theta_\alpha(x^k)| > \varepsilon$. Thus, from Lemma 11, we obtain

$$\begin{aligned}(N+1)\varepsilon < (N+1)\min\{|\theta_\alpha(x^k)| \mid k = 0, \ldots, N\} &\leq \sum_{k=0}^{N}|\theta_\alpha(x^k)|\\ &\leq c\sum_{k=0}^{N}\left(F_{j_k^*}(x^k) - F_{j_k^*}(x^{k+1})\right).\end{aligned}$$

Hence, since by Lemma 5 we have $F(x^{k+1}) \preceq F(x^k)$, we then conclude that

$$\begin{aligned}(N+1)\varepsilon < c\sum_{j=1}^{m}\sum_{k=0}^{N}\left(F_j(x^k) - F_j(x^{k+1})\right) &= c\sum_{j=1}^{m}\left(F_j(x^0) - F_j(x^{N+1})\right)\\ &\leq c\sum_{j=1}^{m}\left(F_j(x^0) - F_j(\bar{x})\right),\end{aligned}$$

where $\bar{x}$ is as in Theorem 6. Therefore,

$$N + 1 < \frac{c \sum_{j=1}^{m} \big(F_j(x^0) - F_j(\bar{x})\big)}{\varepsilon},$$

concluding the proof. $\square$

We end this section by noting that the computational cost associated with each iteration k of the MPG-Explicit Algorithm includes solving the subproblem at Step 1 and computing the stepsize t_k at Step 3. In terms of the line search procedure, the computational cost essentially involves evaluations of G_j functions at different trial points. The following remark outlines the maximum number of function evaluations needed to compute t_k.

Remark 2. *For iteration k, let $\omega(k) \geq 0$ denote the number of backtracking iterations in the line search procedure to calculate t_k, i.e., the number of t_{new} stepsizes computed. Thus, $t_k \leq \tau_2^{\omega(k)}$. Since, by Lemma 9, $t_k \geq t_{\min}$, we obtain $\log(t_{\min}) \leq \omega(k)\log(\tau_2)$. Considering that $\tau_2 < 1$, it follows that $\omega(k) \leq \log(t_{\min})/\log(\tau_2)$. Consequently, the line search procedure involves at most:*

- *$1 + m\omega(k) \leq 1 + m\log(t_{\min})/\log(\tau_2)$ evaluations of G_j's functions at Steps 3.1 and 3.3;*
- *$m - 1$ evaluations of H_j's functions at Step 3.2.*

6 Numerical results

This section provides numerical results to assess the practical performance of the MPG-Explicit algorithm. We are mainly interested in verifying the effectiveness of using the new explicit line search procedure in a proximal-gradient-type method. For this purpose, we consider the following methods in the reported tests:

- MPG-Explicit: The multiobjective proximal gradient algorithm with the explicit line search procedure proposed in Section 3.1.
- MPG-Armijo: The *vanilla* proximal gradient algorithm with Armijo line search proposed in [60]. The stepsize is defined as

  $$t_k =: \max_{\ell \in \mathbb{N}} \{2^{-\ell} \mid F_j(x^k + 2^{-\ell} d^k) \leq F_j(x^k) - \sigma 2^{-\ell} \psi_{x^k}(p^k),\ j = 1, \dots, m\},$$

  where $\sigma := 10^{-4}$ is an algorithmic parameter.
- MPG-Accelerated: The accelerated proximal gradient method proposed in [62]. Under assumption (A2), the algorithm starts with $x^0 = y^1 \in \operatorname{dom}(F)$, $\alpha \geq L_{\max} := \max_{j=1,\dots,m} L_j$, and updates the iterates as follows:

  $$x^k := \arg\min_{u \in \mathbb{R}^n} \max_{i=1,\dots,m} \big[\nabla G_i(y^k)^\top (u - y^k) + H_i(u) + G_i(y^k) - F_i(x^{k-1})\big] + \frac{\alpha}{2}\|u - y^k\|^2,$$

where the auxiliary variable y^k is updated with a momentum term $y^{k+1} := x^k + \gamma_k(x^k - x^{k-1})$, with $\gamma_k := (t_k - 1)/t_{k+1}$ and $t_{k+1} := \sqrt{t_k^2 + 1/4} + 1/2$. In our implementation, the parameter α is updated iteratively using an adaptive backtracking procedure as described in [62, Remark 3].

- MPG-Normal: The so-called *normal* proximal gradient method described in [62]. This algorithm essentially coincides with the MPG-Accelerated algorithm with $y^k = x^{k-1}$ for all $k \geq 1$.
- MPG-Implicit: The *implicit* proximal gradient algorithm proposed in [10]. This algorithm can be viewed as a precursor to the MPG-Explicit algorithm. It is similar to the MPG-Explicit algorithm, with the line search procedure replaced by the following scheme: Beginning with $\alpha = 1$, if

$$G_j(p_\alpha(x^k)) \leq G_j(x^k) + \nabla G_j(x^k)^\top d^k + \frac{1}{2\alpha}\|d^k\|^2, \quad j = 1, \ldots, m,$$

then $x^{k+1} := p_\alpha(x^k)$, and a new iteration is initialized. Otherwise, set $\alpha \leftarrow \alpha/2$ and solve the proximal subproblem (8) again.

The success stopping criterion for all algorithms is defined as:

$$\frac{\|x^k - y^k\|_\infty}{\max\{1, \|y^k\|_\infty\}} \leq 10^{-4}. \tag{24}$$

For the MPG-Accelerated algorithm, y^k is updated using the momentum term as described in the algorithm's formulation. In the other algorithms, y^k is taken as x^{k-1}. This stopping criterion is based on the approach used in [62].

The maximum number of allowed iterations is set to 200, after which the algorithm is considered to have failed. The experiments were conducted in MATLAB version 24.2 (R2024b), on a computer with a 3.7 GHz Intel Core i5 6-Core processor and 8GB 2667MHz DDR4 RAM, running macOS Sequoia 15.0. All codes are available at https://github.com/lfprudente/MPG.

6.1 Implementation details of MPG-Explicit

The algorithmic parameters for the MPG-Explicit implementation are set as follows: $\alpha = 1$, $\gamma = 1.9999$, $\tau_1 = 0.1$, and $\tau_2 = 0.9$. Given that the explicit line search procedure utilizes only information from the differentiable functions G_j, we exploit this characteristic by implementing the backtracking scheme based on quadratic polynomial interpolations of G_j's functions.

The selection of t_{new} in Step 3 is performed using an adaptive procedure that considers a reference function G_{j_k}. Specifically:

- In Step 3.1, the reference function is $G_{j_k^*}$, where $j_k^* \in \operatorname{argmax}_{j=1,\ldots,m} \nabla G_j(x^k)^\top d^k$.
- In Step 3.3, the reference function is G_{j_k}, corresponding to a function for which the inequality in Step 3.3 is not satisfied. According to Lemma 4 , the inequality in Step 3.3 is violated for G_{j_k} if $F_{j_k}(x^k + t_{\text{trial}} d^k) > F_{j_k}(x^k)$, as indicated in Step 3.2.

To update t_{new}, we proceed as follows. Let G_{j_k} be the reference function and define $\varphi(t) := G_{j_k}(x^k + td^k)$. If $\varphi'(0) = \nabla G_{j_k}(x^k)^\top d^k < 0$, we compute t_q, the minimizer of the quadratic approximation $\varphi_q(t)$, given by

$$t_q = -\frac{\varphi'(0)t_{\text{trial}}^2}{2\left[\varphi(t_{\text{trial}}) - \varphi(0) - \varphi'(0)t_{\text{trial}}\right]},$$

where $\varphi_q(t)$ interpolates the values $\varphi_q(0) = \varphi(0)$, $\varphi_q'(0) = \varphi'(0)$, and $\varphi_q(t_{\text{trial}}) = \varphi(t_{\text{trial}})$. If $t_q \in [\tau_1 t_{\text{trial}}, \tau_2 t_{\text{trial}}]$, we set $t_{\text{new}} = t_q$. Otherwise, we set $t_{\text{new}} = t_{\text{trial}}/2$. The pseudocode for computing t_{new} is given below:

Computation of t_{new}

```
1:  **Input:** $t_{\text{trial}}, d^k, x^k, \tau_1, \tau_2$
2:  Choose reference function $G_{j_k}$
3:  Define $\varphi(t) := G_{j_k}(x^k + td^k)$
4:  **if** $\varphi'(0) = \nabla G_{j_k}(x^k)^\top d^k < 0$ **then**
5:      Compute $t_q$, the minimizer of the quadratic approximation $\varphi_q(t)$
6:      **if** $t_q \in [\tau_1 t_{\text{trial}}, \tau_2 t_{\text{trial}}]$ **then**
7:          Set $t_{\text{new}} = t_q$
8:      **else**
9:          Set $t_{\text{new}} = t_{\text{trial}}/2$
10:     **end if**
11: **else**
12:     Set $t_{\text{new}} = t_{\text{trial}}/2$
13: **end if**
14: **Output:** $t_{\text{new}}$
```

6.2 Test problems

The chosen test problems are related to robust multiobjective optimization (see [16]). Robust optimization deals with uncertainty in the data of optimization problems, requiring the optimal solution to occur in the worst possible scenario, i.e., for the most adverse values that the uncertain data may take.

In the following, we discuss how the test problems are designed (see [5]). The differentiable components G_j come from convex multiobjective problems found in the literature. Table 1 provides the main characteristics of the selected problems, including the problem name, the corresponding reference for its formulation, the number of variables (n), and the number of objectives (m). For each test problem, we assume that $\text{dom}(H_j) = \{x \in \mathbb{R}^n \mid lb \preceq x \preceq ub\}$ for $j = 1, \ldots, m$, where $lb, ub \in \mathbb{R}^n$ are given in the last columns of the table. Thus, $\text{dom}(F)$ also coincides with this box.

For a given problem in Table 1, we denote the *uncertainty parameter* by $z \in \mathbb{R}^n$, and assume that the objective functions are as follows:

$$F_j(x) := G_j(x) + x^\top z, \quad x \in \text{dom}(F), \quad j = 1, \ldots, m.$$

Problem	Ref.	n	m	lb	ub
AP1	[3]	2	3	$(-10,-10)$	$(10,10)$
AP2	[3]	1	2	-100	100
AP4	[3]	3	3	$(-10,-10,-10)$	$(10,10,10)$
BK1	[43]	2	2	$(-5,-5)$	$(10,10)$
DGO2	[43]	1	2	-9	9
FDS	[33]	5	3	$(-2,\ldots,-2)$	$(2,\ldots,2)$
IKK1	[43]	2	3	$(-50,-50)$	$(50,50)$
JOS1	[45]	100	2	$(-100,\ldots,-100)$	$(100,\ldots,100)$
Lov1	[46]	2	2	$(-10,-10)$	$(10,10)$
MGH33	[49]	10	10	$(-1,\ldots,-1)$	$(1,\ldots,1)$
MHHM2	[43]	2	3	$(0,0)$	$(1,1)$
MOP7	[43]	2	3	$(-400,-400)$	$(400,400)$
PNR	[54]	2	2	$(-2,-2)$	$(2,2)$
SD	[59]	4	2	$(1,\sqrt{2},\sqrt{2},1)$	$(3,3,3,3)$
SLCDT2	[58]	10	3	$(-1,\ldots,-1)$	$(1,\ldots,1)$
SP1	[43]	2	2	$(-100,-100)$	$(100,100)$
Toi4	[49]	4	2	$(-2,-2,-2,-2)$	$(5,5,5,5)$
Toi8	[49]	3	3	$(-1,-1,-1,-1)$	$(1,1,1,1)$
VU2	[43]	2	2	$(-3,-3)$	$(3,3)$
ZDT1	[67]	30	2	$(0,\ldots,0)$	$(1,\ldots,1)$
ZLT1	[43]	10	5	$(-1000,\ldots,-1000)$	$(1000,\ldots,1000)$

Table 1: List of test problems.

Minimizing $F_j(x)$ under the worst-case scenario involves solving

$$\min_{x\in\mathrm{dom}(F)} G_j(x) + \max_{z\in\mathcal{Z}_j} x^\top z,$$

where $\mathcal{Z}_j \subset \mathbb{R}^n$ is the *uncertainty set*. Thus, we define $H_j : \mathrm{dom}(F) \to \mathbb{R}$ as

$$H_j(x) := \max_{z\in\mathcal{Z}_j} x^\top z, \quad j = 1,\ldots,m. \tag{25}$$

Assuming that $\mathcal{Z}_j$ is a nonempty and bounded polyhedron given by $\mathcal{Z}_j = \{z \in \mathbb{R}^n \mid A_j z \preceq b_j\}$, where $A_j \in \mathbb{R}^{d\times n}$ and $b_j \in \mathbb{R}^d$, we can express (25) as

$$\begin{array}{ll} \max_z & x^\top z \\ \text{s.t.} & A_j z \preceq b_j. \end{array} \tag{26}$$

This means that evaluating $H_j(\cdot)$ requires solving a linear programming problem.

Let us discuss how the subproblem can be solved. It is easy to see that (8) can be reformulated by introducing an extra variable $\tau \in \mathbb{R}$ as follows

$$\begin{array}{ll} \min_{\tau,u} & \tau + \frac{1}{2\alpha}\|u - x^k\|^2 \\ \text{s.t.} & \nabla G_j(x^k)^\top (u - x^k) + H_j(u) - H_j(x^k) \leq \tau, \quad j = 1,\ldots,m, \\ & lb \preceq u \preceq ub. \end{array} \tag{27}$$

Now, by noting that the dual problem of (26) is given by

$$\begin{array}{ll} \min_{w_j} & b_j^\top w_j \\ \text{s.t.} & A_j^\top w_j = x, \\ & w_j \succeq 0, \end{array}$$

we can use the strong duality property to reformulate (27) as the following quadratic programming problem:

$$\begin{array}{ll} \min_{\tau,u,w_j} & \tau + \frac{1}{2\alpha}\|u - x^k\|^2 \\ \text{s.t.} & \nabla G_j(x^k)^\top (u - x^k) + b_j^\top w_j - H_j(x^k) \leq \tau, \quad j = 1, \ldots, m, \\ & A_j^\top w_j = u, \quad j = 1, \ldots, m, \\ & w_j \succeq 0, \quad j = 1, \ldots, m, \\ & lb \preceq u \preceq ub. \end{array} \tag{28}$$

For more details, see [60, Section 5.2 (a)]. In the codes, we use a simplex-dual method (*linprog* routine) to solve (26), and an interior point method (*quadprog* routine) to solve (28). We mention that if *linprog* or *quadprog* fail to solve (26) or (28), respectively, the execution of the main algorithm is terminated, and the algorithm is considered to have failed in solving the original problem.

In our experiments, we define the uncertainty set $\mathcal{Z}_j$ as follows: Let $B_j \in \mathbb{R}^{n\times n}$ be a (random) nonsingular matrix, and $\delta > 0$ be a given parameter. Then

$$\mathcal{Z}_j := \{z \in \mathbb{R}^n \mid -\delta e \preceq B_j z \preceq \delta e\}, \quad j = 1, \ldots, m,$$

where $e = (1, \ldots, 1)^T \in \mathbb{R}^n$. The parameter δ plays a crucial role in controlling the uncertainty of the problem and is defined as:

$$\delta := \hat{\delta}\|\hat{x}\|,$$

where $\hat{\delta}$ is randomly chosen from the interval $[0.02, 0.10]$, and $\hat{x}$ is an arbitrary point in $\text{dom}(F)$.

6.3 Efficiency and robustness

For each test problem, we employed 100 randomly generated starting points within the corresponding set $\text{dom}(F)$. In this phase, each combination of a problem and a starting point was treated as an independent instance and solved by the five algorithms. A run is considered successful if (24) is satisfied, regardless of the objective function value. The results in Figure 1 are presented using performance profiles [29] and compare the algorithms with respect to: (a) CPU time; (b) number of iterations; (c) number of evaluations of H_j's functions. It is worth noting that, in a performance profile graph, the extreme left (at 1 in the domain) assesses *efficiency*, while the extreme right evaluates *robustness*.

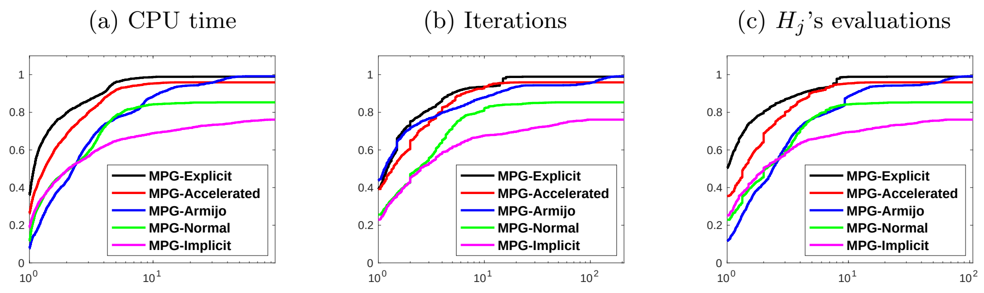

Fig. 1: Performance profiles considering: (a) CPU time; (b) number of iterations; (c) number of evaluations of H_j's functions.

As shown in Figure 1 (a), with respect to CPU time, the MPG-Explicit algorithm is the fastest among the tested methods, followed by MPG-Accelerated and then the remaining algorithms. This aligns with additional results, indicating that MPG-Explicit, MPG-Accelerated, and MPG-Armijo are competitive in terms of iteration count, outperforming MPG-Normal and MPG-Implicit, see Figure 1 (b). However, as shown in Figure 1 (c), MPG-Explicit further distinguishes itself by requiring fewer evaluations of H_j's functions, making it the fastest method overall. Concerning robustness, both MPG-Explicit and MPG-Armijo exhibit high robustness values, with success rates of 98.9% and 99.1%, respectively, indicating their reliability across the set of test problems. MPG-Accelerated also shows a strong robustness score of 95.9%, while MPG-Normal and MPG-Implicit achieve lower robustness, with success rates of 85.3% and 76.1%, respectively. Table 2 summarizes the efficiency and robustness scores for each algorithm.

	Efficiency (%)			Robustness (%)
	Time	It.	H_j	
MPG-Explicit	36.0	39.2	50.4	98.9
MPG-Accelerated	26.2	39.2	35.5	95.9
MPG-Armijo	7.7	43.8	11.8	99.1
MPG-Normal	11.6	25.7	22.9	85.3
MPG-Implicit	18.8	23.0	25.2	76.1

Table 2: Efficiency and robustness of the algorithms on the chosen set of test problems.

Table 3 complements the findings by displaying, for each problem, the percentage of successfully solved instances and the average values of each metric (CPU time, number of iterations, and number of evaluations of H_j's functions) for each algorithm, with the best results highlighted in bold. In line with the performance profiles, the table shows that the MPG-Explicit, MPG-Accelerated, and MPG-Armijo algorithms exhibit high robustness, with success rates close to 100%. Failures across all methods primarily resulted from reaching the maximum iteration limit or errors in solving the subproblem via the *quadprog* routine. For iteration count, MPG-Explicit, MPG-Accelerated, and MPG-Armijo alternate as the leading methods across different problems, reflecting their competitive balance in iteration efficiency. On the other hand, MPG-Explicit consistently delivers the best outcomes in terms of CPU time and H_j evaluations, highlighting the potential benefits of the new explicit line search procedure, particularly in cases where evaluating the H_j functions is computationally expensive.

6.4 Pareto frontiers

In multiobjective optimization, the main goal is to estimate the Pareto frontier of a given problem. A commonly employed strategy for this task involves running an algorithm from various starting points and collecting the Pareto optimal points found. Therefore, for a given test problem, we execute each algorithm for 2 minutes to obtain an approximation of the Pareto frontier. As shown in Table 3, this 2-minute limit allows for multiple restarts of the algorithms from different initial points, providing an adequate estimate of the Pareto frontier. Moreover, this methodology favors faster algorithms, enabling them to explore more starting points within the fixed time, which typically results in a larger set of Pareto optimal points.

We compare the results using well-known metrics such as *Purity*, (Γ and Δ) *Spread*, and *Hypervolume*. In essence, for a given problem, the Purity metric measures the algorithm's ability to discover points on the Pareto frontier, while the Spread metric assesses its capability to obtain well-distributed points along the Pareto frontier. The Hypervolume metric, on the other hand, quantifies the volume in the objective space covered by the Pareto optimal points found by the algorithm, relative to a predefined reference point. For each problem, we set the reference point as the maximum value observed for each objective function across all solutions from all algorithms and initial points. For a detailed discussion of these metrics and their uses along with performance profiles, see [28, 62].

	MPG-Explicit				MPG-Accelerated				MPG-Armijo				MPG-Normal				MPG-Implicit			
Problem	%	Time	It.	H_j	%	Time	It.	H_j	%	Time	It.	H_j	%	Time	It.	H_j	%	Time	It.	H_j
AP1	99	1.10	56.1	173.6	97	**0.70**	**34.5**	**114.2**	**100**	1.29	56.9	211.6	66	1.46	75.7	238.2	97	1.82	91.2	276.7
AP2	**100**	**0.05**	**2.6**	**7.1**	**100**	0.07	2.9	10.1	**100**	0.42	23.4	63.8	**100**	0.08	3.8	12.0	49	1.24	73.8	149.7
AP4	96	1.07	53.5	169.4	**99**	**0.65**	**29.9**	**103.2**	97	1.25	57.0	198.2	61	1.12	55.9	178.4	84	2.27	110.0	332.9
BK1	**100**	**0.08**	**4.5**	**11.0**	**100**	0.16	9.1	23.3	**100**	0.38	19.4	57.4	**100**	0.15	8.4	21.9	89	0.16	9.4	20.8
DGO2	**100**	**0.22**	15.8	**33.6**	**100**	0.23	**15.5**	35.2	**100**	0.50	15.8	81.2	**100**	0.24	17.3	36.7	**100**	**0.22**	16.3	34.6
FDS	**100**	0.95	42.0	153.1	**100**	0.44	19.9	71.7	**100**	0.85	26.5	141.5	**100**	1.10	54.4	173.7	**100**	**0.35**	**17.7**	**56.2**
IKK1	99	**0.38**	**17.5**	**56.4**	**100**	0.41	18.9	62.5	99	0.51	**17.5**	80.2	**100**	0.87	41.3	129.7	99	1.45	66.2	201.5
JOS1	**100**	**0.04**	**2.0**	**6.0**	**100**	0.07	3.3	9.0	**100**	0.29	**2.0**	46.6	**100**	0.07	3.8	10.1	41	2.16	122.6	247.1
Lov1	**100**	**0.10**	**5.7**	**13.3**	**100**	0.21	11.3	28.5	**100**	0.33	15.7	45.4	**100**	0.20	10.3	26.5	39	0.35	20.1	42.3
MGH33	90	1.09	48.9	151.2	79	2.43	105.3	334.1	**94**	**1.03**	**37.8**	**146.0**	18	1.68	76.8	242.3	92	1.23	55.9	170.7
MHHM2	**100**	**0.12**	3.2	**13.0**	**100**	0.21	5.6	24.5	**100**	0.30	**2.7**	36.9	**100**	0.28	7.9	31.8	**100**	0.15	4.8	17.3
MOP7	**100**	0.51	20.0	63.3	**100**	**0.36**	**14.4**	**46.3**	**100**	0.56	20.0	69.3	**100**	0.51	20.0	63.0	2	4.58	185.5	559.5
PNR	**100**	0.22	9.8	23.8	**100**	**0.19**	**7.5**	**21.9**	**100**	0.62	17.4	73.9	**100**	0.25	10.3	27.5	**100**	0.24	11.8	25.7
SD	**100**	0.36	18.1	38.3	**100**	**0.32**	**16.2**	**34.6**	**100**	0.36	18.1	38.3	**100**	0.40	20.0	42.4	**100**	0.36	18.1	38.3
SLCDT2	**100**	**0.35**	13.2	**47.6**	**100**	0.63	24.1	84.3	**100**	0.40	**10.3**	56.3	**100**	0.62	24.1	83.3	0	-	-	-
SP1	**100**	**0.33**	19.0	**41.4**	**100**	0.42	23.5	54.9	**100**	0.60	**18.6**	81.0	**100**	0.95	55.8	119.0	7	3.03	161.1	324.3
Toi4	99	**0.51**	**35.1**	**72.4**	**100**	0.56	37.4	78.7	99	0.57	**35.1**	81.3	90	1.38	95.3	194.5	**100**	0.83	56.9	115.8
Toi8	94	1.38	59.4	195.2	**100**	1.80	82.0	261.9	93	1.56	45.5	249.7	37	1.47	65.4	205.7	**100**	**0.59**	**28.1**	**87.3**
VU2	**100**	**0.08**	**4.6**	**11.3**	**100**	0.10	5.7	14.6	**100**	0.17	**4.6**	26.7	**100**	**0.08**	4.8	11.7	**100**	**0.08**	4.8	11.5
ZDT1	**100**	2.22	96.2	194.4	39	**1.49**	**59.3**	**125.0**	**100**	2.29	93.8	204.0	19	2.80	114.0	230.9	**100**	2.13	93.8	189.5
ZLT1	**100**	0.52	14.8	78.8	**100**	0.26	5.3	38.6	**100**	4.16	117.8	612.2	**100**	0.24	5.1	37.1	99	**0.12**	**1.0**	**10.0**

Table 3: Performance of the algorithms on the chosen set of test problems.

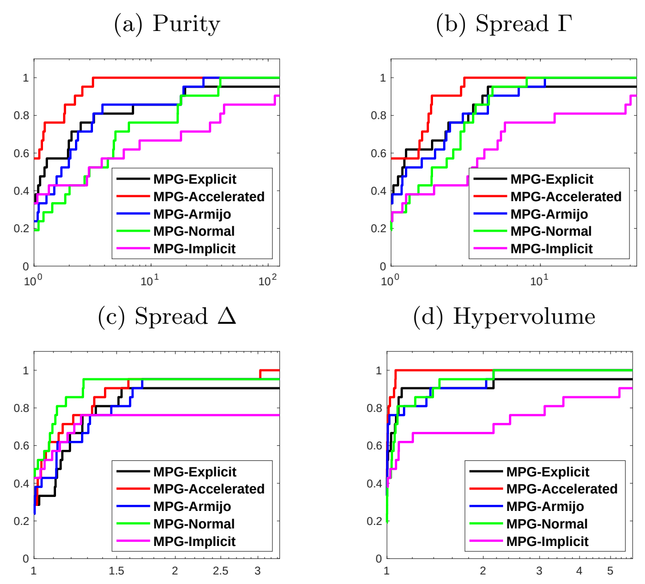

Fig. 2: Metric performance profiles: (a) Purity; (b) Spread Γ; (c) Spread Δ; (d) Hypervolume.

Figure 2 shows that MPG-Explicit remains competitive across all metrics, consistently achieving strong performance. MPG-Accelerated stands out as the most efficient algorithm overall, while MPG-Implicit consistently ranks as the least efficient. For the Purity and Spread Γ metrics, MPG-Accelerated leads, with MPG-Explicit closely following as the second-best, highlighting its ability to effectively discover and distribute points along the Pareto frontier. In the Spread Δ metric, all algorithms perform similarly, except for MPG-Implicit, which ranks significantly lower; surprisingly, MPG-Normal performs slightly better in this case. For the Hypervolume metric, MPG-Accelerated holds a slight advantage, with MPG-Explicit maintaining a competitive position alongside other methods. These results support the conclusion that incorporating the new line search procedure does not compromise the practical performance of a proximal-gradient-type method, especially regarding solution quality.

We mention that the observed results align with the theoretical convergence properties of MPG-Accelerated, which benefits from an accelerated functional convergence rate of $\mathcal{O}(1/k^2)$. This rate likely enhances its effectiveness in estimating the Pareto frontier but does not necessarily translate to faster convergence in runtime. Indeed, the experiments in Section 6.3 confirm this distinction, as MPG-Explicit emerges as the fastest algorithm overall.

For illustrative purposes, Figure 3 below displays the image space along with the approximation of the Pareto frontier obtained by the MPG-Explicit algorithm for the BK1, Lov1, PNR, and VU2 problems. In the graphics, the final iterate is marked by a black dot, while the intermediate iterates appear as blue dots linked by line segments that trace the path from the initial point to the solution. As can be seen, the algorithm visually achieves a satisfactory outline of the Pareto frontiers.

7 Conclusion

We presented a proximal gradient method with a new explicit line search procedure designed for convex multiobjective optimization problems, where each objective function F_j is of the form $F_j := G_j + H_j$, with G_j and H_j being convex functions and G_j being continuously differentiable. The algorithm requires solving only one proximal subproblem per iteration, with the backtracking scheme exclusively applied to the differentiable functions G_j. Our numerical experiments revealed that the proposed method exhibits a reduced number of evaluations of H_j when compared to other proximal gradient algorithms, including the accelerated proximal gradient method and the proximal gradient method with Armijo line search proposed in [62] and [60], respectively. This characteristic makes our approach particularly promising in applications where the computation of H_j is computationally expensive. Some directions for future research include the development of a hybrid strategy that combines our inexpensive G-based test with occasional full evaluations of the composite function F, as well as integrating the proposed line search procedure with other optimization methods, such as Newton-type algorithms.

Declarations

Funding
Y. Bello-Cruz was partially supported by the National Science Foundation (NSF), Grant DMS-2307328; J. G. Melo was partially supported by Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), Grants 312223/2022-6 and 405349/2021-1; L. F. Prudente was partially supported by CNPq, Grants 307294/2023-4 and 405349/2021-1.

Data availability statement
The codes supporting the numerical experiments are freely available in the Github repository, https://github.com/lfprudente/MPG.

Conflicts of interest
The authors declare that they have no conflict of interest.

References

[1] M. A. T. Ansary and J. Dutta. A Proximal Gradient Method for Multi-objective Optimization Problems using Bregman Functions. *Optimization Online*, 2022.

[2] Ya.I. Alber, A.N. Iusem and M.V. Solodov. On the projected subgradient method for nonsmooth convex optimization in a Hilbert space. *Math. Program.*, 81, 23-35, 1998.

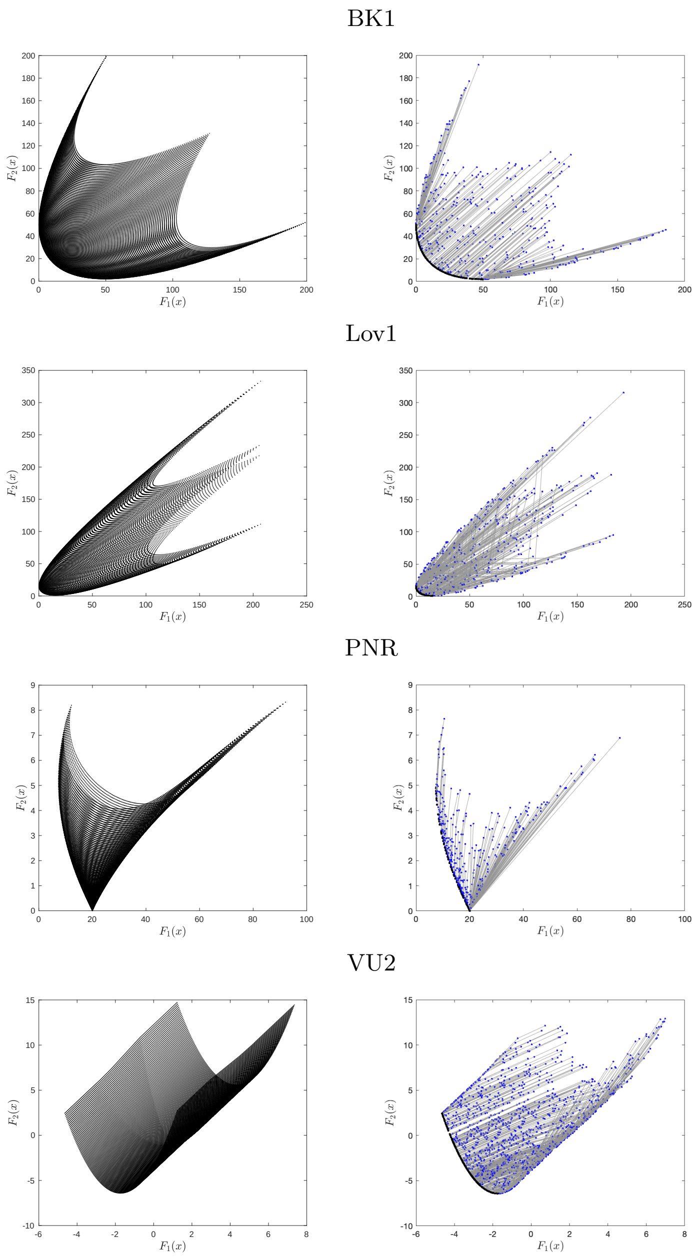

Fig. 3: Image spaces and approximations of the Pareto frontiers obtained by the MPG-Explicit algorithm for the BK1, Lov1, PNR, and VU2 problems.

[3] M. A. T. Ansary. A Newton-type proximal gradient method for nonlinear multi-objective optimization problems. *Optim. Methods Softw.*, 38(3), 570–590, 2023.

[4] P. B. Assunção, O. P. Ferreira, and L. F. Prudente. Conditional gradient method for multiobjective optimization. *Comput. Optim. Appl.*, 78(3), 741–768, 2021.

[5] P. B. Assunção, O. P. Ferreira, and L. F. Prudente. A generalized conditional gradient method for multiobjective composite optimization problems. *Optimization*, 74(2), 473–503, 2023.

[6] A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM J. Imaging Sci.*, 2(1), 183–202, 2009.

[7] J. Y. Bello Cruz. A Subgradient Method for Vector Optimization Problems. *SIAM J. Optim.*, 23(4), 2169-2182, 2013.

[8] J. Y. Bello Cruz and T. T. Nghia. On the convergence of the forward-backward splitting method with linesearches. *Optim. Methods Softw.*, 31(6), 1209–1238, 2016.

[9] J. Y. Bello Cruz, L. R. Lucambio Pérez, and J. G. Melo. Convergence of the projected gradient method for quasiconvex multiobjective optimization. *Nonlinear Anal.*, 74(16), 5268–5273, 2011.

[10] J. Y. Bello Cruz, J. G. Melo, and R. V. G. Serra. A proximal gradient splitting method for solving convex vector optimization problems. *Optimization*, 71(1), 33–53, 2022.

[11] G. C. Bento, O. P. Ferreira and P. R. Oliveira. Unconstrained steepest descent method for multicriteria optimization on Riemannian manifolds. J. Optim. Theory Appl. 154(1), 88–107, 2012.

[12] G. C. Bento and J. X. Cruz Neto. A Subgradient method for multiobjective optimization on Riemannian manifolds. J. Optim. Theory Appl. 159, 125–137, 2013.

[13] G. C. Bento, J. X. Cruz Neto, and A. Soubeyran. A proximal point-type method for multicriteria optimization. *Set-Valued Var. Anal.*, 22(3), 557–573, 2014.

[14] G. C. Bento, O. P. Ferreira, and Y. R. L. Pereira. Proximal point method for vector optimization on Hadamard manifolds. *Oper. Res. Lett.*, 46(1), 13–18, 2018.

[15] D. P. Bertsekas. *Nonlinear programming.* Athena Scientific Optimization and Computation Series. Athena Scientific, Belmont, MA, second edition, 1999.

[16] D. Bertsimas, D. B. Brown, and C. Caramanis. Theory and applications of robust optimization. *SIAM Rev.*, 53(3), 464–501, 2011.

[17] H. Bonnel, A. N. Iusem, and B. F. Svaiter. Proximal methods in vector optimization. *SIAM J. Optim.*, 15(4), 953–970, 2005.

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

[19] L. C. Ceng and J. C. Yao. Approximate proximal methods in vector optimization. *Eur. J. Oper. Res.*, 183(1), 1–19, 2007.

[20] K. Chen, E. H. Fukuda and N. Yamashita. A proximal gradient method with Bregman distance in multi-objective optimization. Pac. J. Optim., 20(4), 809-826, 2024.

[21] J. Chen, L. Tang, and X. Yang. Convergence rates analysis of Interior Bregman Gradient Method for Vector Optimization Problems. *arXiv:2206.10070*, 2022.

[22] J. Chen, X. X. Jiang, L. P. Tang, and X. M. Yang. On the convergence of Newton-type proximal gradient method for multiobjective optimization problems. Optim.

Methods Softw., 40(3), 509–524, 2025.
[23] J. Chen, L. Tang, and X. Yang. Barzilai-borwein proximal gradient methods for multiobjective composite optimization problems with improved linear convergence. *arXiv 2306.09797*, 2023.
[24] T. D. Chuong. Generalized proximal method for efficient solutions in vector optimization. *Numer. Funct. Anal. Optim.*, 32(8), 843–857, 2011.
[25] T. D. Chuong and J. C. Yao. Steepest descent methods for critical points in vector optimization problems. *Appl. Anal.*, 91(10), 1811–1829, 2012.
[26] T. D. Chuong, B. S. Mordukhovich, and J. C. Yao. Hybrid approximate proximal algorithms for efficient solutions in vector optimization. *J. Nonlinear Convex Anal.*, 12(2), 257–285, 2011.
[27] J.X. Da Cruz Neto, G.J.P. Da Silva, O.P. Ferreira, J.O. Lopes. A subgradient method for multiobjective optimization. *Comput. Optim. Appl.*, 54(3), 461–472, 2013.
[28] A. L. Custódio, J. F. A. Madeira, A. I. F. Vaz, and L. N. Vicente. Direct multisearch for multiobjective optimization. *SIAM J. Optim.*, 21(3), 1109–1140, 2011.
[29] E. D. Dolan and J. J. Moré. Benchmarking optimization software with performance profiles. *Math. Program.*, 91(2), 201–213, 2002.
[30] Yu. M. Ermoliev. On the method of generalized stochastic gradients and quasi-Fejér sequences. *Cybernetics*. **5**, 208-220, 1969.
[31] J. Fliege. An Efficient Interior-Point Method for Convex Multicriteria Optimization Problems. Math. Oper. Res. 31(4), 825–845, 2006.
[32] J. Fliege and B. F. Svaiter. Steepest descent methods for multicriteria optimization. *Math. Methods Oper. Res.*, 51(3), 479–494, 2000.
[33] J. Fliege, L. M. Graña Drummond, and B. F. Svaiter. Newton's method for multiobjective optimization. *SIAM J. Optim.*, 20(2), 602–626, 2009.
[34] J. Fliege, A. I. F. Vaz and Vicente, L. N. Complexity of gradient descent for multiobjective optimization. *Optim. Meth. and Soft.*, 34(5), 949–959, 2018.
[35] E. H. Fukuda and L. M. Graña Drummond. On the convergence of the projected gradient method for vector optimization. *Optimization*, 60(8-9), 1009–1021, 2011.
[36] E. H. Fukuda and L. M. Graña Drummond. Inexact projected gradient method for vector optimization. *Comput. Optim. Appl.*, 54(3), 473–493, 2013.
[37] A. G. Gebrie and E. H. Fukuda. Adaptive generalized conditional gradient method for multiobjective optimization. *J. Optim. Theory Appl.*, 206(1):13, 2025.
[38] M. L. N. Gonçalves and L. F. Prudente. On the extension of the Hager–Zhang conjugate gradient method for vector optimization. *Comput. Optim. Appl.*, 76 (3), 889–916, 2020.
[39] M. L. N. Gonçalves, F. S. Lima, and L. F. Prudente. Globally convergent Newton-type methods for multiobjective optimization. *Comput. Optim. Appl.*, 83(2), 403–434, 2022.
[40] L. M. Graña Drummond and A. N. Iusem. A projected gradient method for vector optimization problems. *Comput. Optim. Appl.*, 28(1), 5–29, 2004.
[41] L. M. Graña Drummond and B. F. Svaiter. A steepest descent method for vector optimization. *J. Comput. Appl. Math.*, 175(2), 395–414, 2005.

[42] L. M. Graña Drummond , F. M. P. Raupp and B. F. Svaiter. A quadratically convergent Newton method for vector optimization. Optimization, 63(5), 661–677, 2012.

[43] S. Huband, P. Hingston, L. Barone, and L. While. A review of multiobjective test problems and a scalable test problem toolkit. *IEEE Trans. Evol. Comput.*, 10(5), 477–506, 2006.

[44] X. Jiang. A nonmonotone proximal quasi-Newton method for multiobjective optimization. *arXiv 2310.01751*, 2023.

[45] Y. Jin, M. Olhofer, and B. Sendhoff. Dynamic weighted aggregation for evolutionary multi-objective optimization: Why does it work and how? In *Proceedings of the 3rd Annual Conference on Genetic and Evolutionary Computation*, GECCO'01, page 1042–1049, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc. ISBN 1558607749.

[46] A. Lovison. Singular continuation: Generating piecewise linear approximations to pareto sets via global analysis. *SIAM J. Optim.*, 21(2), 463–490, 2011.

[47] D. T. Luc. Theory of vector optimization, Lectures Notes in economics and mathematical systems, vol. 319, 1989.

[48] L. R. Lucambio Pérez and L. F. Prudente. Nonlinear conjugate gradient methods for vector optimization. *SIAM J. Optim.*, 28(3), 2690–2720, 2018.

[49] K. Mita, E. H. Fukuda, and N. Yamashita. Nonmonotone line searches for unconstrained multiobjective optimization problems. *J. Global Optim.*, 75(1), 63–90, 2019.

[50] Y. Nishimura, E. H. Fukuda, and N. Yamashita. Monotonicity for multiobjective accelerated proximal gradient methods. J. Oper. Res. Soc. Japan. 67(1), 1–17, 2024.

[51] J.-W. Peng and J. Ren. Proximal quasi-Newton methods for multiobjective optimization problems. *arXiv 2108.00125*, 2022.

[52] J.-W. Peng and J.-C. Yao. The quasi-Newton method for the composite multiobjective optimization problems. *arXiv 2309.04966*, 2023.

[53] Z. Povalej. Quasi-Newton's method for multiobjective optimization. *J. Comput. Appl. Math.*, 255, 765–777, 2014.

[54] M. Preuss, B. Naujoks, and G. Rudolph. Pareto set and EMOA behavior for simple multimodal multiobjective functions. In T. P. Runarsson, H.-G. Beyer, E. Burke, J. J. Merelo-Guervós, L. D. Whitley, and X. Yao, editors, *Parallel Problem Solving from Nature - PPSN IX*, pages 513–522, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg. ISBN 978-3-540-38991-0.

[55] L. F. Prudente and D. R. Souza. A quasi-Newton method with wolfe line searches for multiobjective optimization. *J. Optim. Theory Appl.*, 194(3), 1107–1140, 2022.

[56] S. Qu, C. Liu, M. Goh, Y. Li, and Y. Ji. Nonsmooth multiobjective programming with quasi-Newton methods. *Eur. J. Oper. Res.*, 235(3), 503–510, 2014.

[57] A. I. R. Burachik, L. M. Graña Drummond and B. F. Svaiter. Full convergence of the steepest descent method with inexact line searches. *Optimization*, 32(2), 137–146, 1995.

[58] O. Schütze, M. Laumanns, C. A. Coello Coello, M. Dellnitz, and E.-G. Talbi. Convergence of stochastic search algorithms to finite size Pareto set approximations.

J. Global Optim., 41(4), 559–577, 2008.

[59] W. Stadler and J. Dauer. Multicriteria optimization in engineering: A tutorial and survey. *Progr. Astronaut. Aero.*, 150, 209–209, 1993.

[60] H. Tanabe, E. H. Fukuda, and N. Yamashita. Proximal gradient methods for multiobjective optimization and their applications. *Comput. Optim. Appl.*, 72(2), 339–361, 2019.

[61] H. Tanabe, E. H. Fukuda, and N. Yamashita. A globally convergent fast iterative shrinkage-thresholding algorithm with a new momentum factor for single and multi-objective convex optimization. *arXiv 2205.05262*, 2022.

[62] H. Tanabe, E. H. Fukuda, and N. Yamashita. An accelerated proximal gradient method for multiobjective optimization. *Comput. Optim. Appl.*, 86(2), 421–455, 2023.

[63] H. Tanabe, E. H. Fukuda, and N. Yamashita. Convergence rates analysis of a multiobjective proximal gradient method. *Optim. Lett.*, 17(2), 333–350, 2023.

[64] J. Wang, Y. Hu, C. K. Wai Yu, C. Li, and X. Yang. Extended Newton methods for multiobjective optimization: majorizing function technique and convergence analysis. *SIAM J. Optim.*, 29(3), 2388–2421, 2019.

[65] J. Zhang and X. Yang. The convergence rate of the accelerated proximal gradient algorithm for multiobjective optimization is faster than $\mathcal{O}(1/k^2)$. *arXiv 2312.06913*, 2023.

[66] X. Zhao and R. Raushan and D. Ghosh and J-C. Yao and M Qi. Proximal gradient method for convex multiobjective optimization problems without Lipschitz continuous gradients. *Comput. Optim. Appl.*, 91,27–66, 2025.

[67] E. Zitzler, K. Deb, and L. Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evol. Comput.*, 8(2), 173–195, 2000.