Outi Montonen | Kaisa Joki

Multiobjective double bundle method for nonsmooth constrained multiobjective DC optimization

Turku Centre for Computer Science

TUCS Technical Report
No 1174, January 2017

Multiobjective double bundle method for nonsmooth constrained multiobjective DC optimization

Outi Montonen
University of Turku, Department of Mathematics and Statistics,
FI-20014 Turku, Finland
outi.montonen@utu.fi
Kaisa Joki
University of Turku, Department of Mathematics and Statistics,
FI-20014 Turku, Finland
kaisa.joki@utu.fi

TUCS Technical Report
No 1174, January 2017

Abstract

The multiobjective DC optimization problems arise naturally, for example, in data classification and cluster analysis playing a crucial role in data mining. In this paper, we propose a new multiobjective double bundle method designed for nonsmooth multiobjective optimization problems having objective and constraint functions which can be presented as a difference of two convex (DC) functions. The method is descent and it generalizes the ideas of the double bundle method for multiobjective and constrained problems. We utilize the special cutting plane model angled for the DC improvement function such that the convex and the concave behaviour of the function is captured. The method is proved to be finitely convergent to a weakly Pareto stationary point under mild assumptions. Finally, we consider some numerical experiments and compare the solutions produced by our method with the method designed for general nonconvex multiobjective problems. This is done in order to validate the usage of the method aimed specially for DC objectives instead of general nonconvex method.

Keywords: Multiobjective optimization, Nonsmooth optimization, Nonconvex optimization, DC programming, Bundle methods

TUCS Laboratory
Turku Optimization Group (TOpGroup)

1 Introduction

Multiobjective optimization problems arise naturally in the wide range of practical applications, since the objectives under the scope are usually simultaneously related to various goals. Thus, compromises have to be made in order to obtain a solution being as good as possible for every objective. The real-life applications for multiobjective optimization can be found, for instance, in the fields of economics [32], engineering [29], and mechanics [30], to name but a few. Along with multiobjective nature, many practical applications have nonsmooth (i.e. nondifferentiable) characteristics.

This paper focuses on multiobjective nonsmooth optimization, and the particular interest is in descent methods. The essential feature of a descent method is the ability to obtain a better solution for each objective at every iteration. In literature, there are some nonsmooth descent methods for convex (see e.g. [4, 5, 18]) and for nonconvex (see e.g. [25, 28, 35, 40]) multiobjective problems. A descent method can be used either by running it repeatedly from different starting points and, therefore, obtain an approximation of the set of optimal solutions, or as a component of some interactive method [27, 28, 31].

A wide subclass of nonconvex functions is formed by the functions having special structure such that they can be decomposed as a difference of two convex functions. These functions are called DC functions. The benefit of the DC functions springs from the ability to utilize the convex analysis and the fact that many functions can be expressed as a DC function. The DC decomposition is not unique, and unfortunately, it might be hard to single out. In practice, the problems with objectives in explicit DC form arise, for instance, in clustering [3], spherical separability problems [9], production-transportation planning [12], and wireless sensor network planning [1]. There exists a lot of studies dedicated to the theory of the DC functions (see e.g. [10, 11, 39]) and to develop single-objective methods for the DC objectives from the different bases (see e.g. [7, 13, 15, 16, 20, 21, 33, 37]). However, the DC functions as the objectives of the multiobjective optimization problem has attracted significantly less attention. In [8, 34, 38], there are presented optimality conditions for the multiobjective DC optimization problem. Additionally, few proximal point methods in [14] have lately come to light.

The aim of this paper is to bring together two areas of optimization and to design a new descent multiobjective method with DC objectives being able to handle DC constraints. The new multiobjective double bundle method for DC functions (MDBDC) utilizes the DC structure of the objective and the constraint functions. The method is inspired by the good numerical performance of the single-objective double bundle method for DC functions (DBDC) [16] and its ability to find global solutions although it is only a local method.

The basic idea of MDBDC is to combine the main features of DBDC with the use of the improvement function [18, 40] as, for instance, in the multiobjective proximal bundle method (MPB) [25, 28]. Along with the sketch of the method, we prove the finite convergence of MDBDC to the weakly Pareto stationary solution under mild assumptions. By the authors' best knowledge, there does not exist any other specially for multiobjective DC optimization designed descent method such that weak Pareto stationarity of the solutions can be ensured instead of Pareto criticality. We analyze the numerical performance of MDBDC, and compare the results obtained by MDBDC with the ones obtained by MPB. MPB is used, since it is a method for a problem with general nonconvex objectives having somehow similar structure than our method. The purpose of this comparison is to motivate the use of the method designed specially for the DC objectives instead of the general nonconvex method.

The remainder of the paper is organized as follows. A brief summary of the relevant material on multiobjective and DC optimization is given in Section 2. Section 3 is devoted to derive the new MDBDC method and to prove its convergence. In Section 4, we investigate the numerical properties of MDBDC. Finally, in Section 5 some concluding remarks are given.

2 Preliminaries

We consider a *multiobjective DC optimization problem* of the form

$$\min_{\boldsymbol{x}\in X} \quad f_1(\boldsymbol{x}), \dots, f_k(\boldsymbol{x}), \tag{1}$$

where $X = \{\boldsymbol{x} \in \mathbb{R}^n \,|\, g_l(\boldsymbol{x}) \leq 0,\, l \in L\}$ and $L = \{1, \dots, m\}$. Additionally, the set $I = \{1, \dots, k\}$ denotes the indices of the objectives. The objectives $f_i : \mathbb{R}^n \to \mathbb{R}$, $i \in I$ and the constraints $g_l : \mathbb{R}^n \to \mathbb{R}$, $l \in L$ are assumed to be *DC functions*. A function f is a DC function if it can be decomposed as a difference of two convex functions $p : \mathbb{R}^n \to \mathbb{R}$ and $q : \mathbb{R}^n \to \mathbb{R}$ such that $f = p - q$. This is called a *DC decomposition* of f, where p and q are *DC components*.

The objectives and the constraints of the problem (1) may be nonsmooth. If a DC function is nonsmooth, then at least one of the DC components is nonsmooth. Based on the DC structure, DC functions are locally Lipschitz continuous at $\boldsymbol{x} \in \mathbb{R}^n$ (LLC) [11] meaning that there exist a *Lipschitz constant* $K > 0$ and $\varepsilon > 0$ such that $|f_i(\boldsymbol{y}) - f_i(\boldsymbol{z})| \leq K\|\boldsymbol{y} - \boldsymbol{z}\|$ for all $\boldsymbol{y}, \boldsymbol{z} \in B(\boldsymbol{x}; \varepsilon)$, where $B(\boldsymbol{x}; \varepsilon)$ is an open ball with a center $\boldsymbol{x}$ and a radius ε.

Next we briefly recall relevant results from nonsmooth, DC and multiobjective optimization. For more details we refer to [2, 6, 11, 24, 27, 39]. We begin with two useful properties of DC functions. First, if f is of the form

$$f(\boldsymbol{x}) = \max\big\{f_j(\boldsymbol{x}) \,\big|\, j \in \mathcal{J}, \mathcal{J} \text{ is finite and } f_j \text{ is a DC function}\big\}, \tag{2}$$

then f is a DC function [11]. Second, for a DC function f, there exists the *direc-*

tional derivative $f'(\boldsymbol{x}; \boldsymbol{d})$ at $\boldsymbol{x} \in \mathbb{R}^n$ in every direction $\boldsymbol{d} \in \mathbb{R}^n$ [11] and

$$f'(\boldsymbol{x}; \boldsymbol{d}) = \lim_{t \downarrow 0} \frac{f(\boldsymbol{x} + t\boldsymbol{d}) - f(\boldsymbol{x})}{t}.$$

Thus, a DC function is said to be *directionally differentiable* at any $\boldsymbol{x}$.

The *subdifferential* of a convex function f at the point $\boldsymbol{x} \in \mathbb{R}^n$ is

$$\partial_c f(\boldsymbol{x}) = \{\boldsymbol{\xi} \in \mathbb{R}^n \mid f(\boldsymbol{y}) \geq f(\boldsymbol{x}) + \boldsymbol{\xi}^T(\boldsymbol{y} - \boldsymbol{x}) \text{ for all } \boldsymbol{y} \in \mathbb{R}^n\}$$

being a nonempty, convex and compact set. The element $\boldsymbol{\xi} \in \partial_c f(\boldsymbol{x})$ is called a *subgradient* of f at $\boldsymbol{x}$. Additionally, for a convex function f and all $\boldsymbol{d} \in \mathbb{R}^n$ at $\boldsymbol{x}$ (see e.g. [2])

$$f'(\boldsymbol{x}; \boldsymbol{d}) = \max\{\boldsymbol{\xi}^T \boldsymbol{d} \mid \boldsymbol{\xi} \in \partial_c f(\boldsymbol{x})\}. \tag{3}$$

We give the following two useful subdifferential calculus rules [2] for convex functions. First, if f is the sum of convex functions f_j, $j \in \mathcal{J}$ such that $\mathcal{J}$ is finite, then

$$f(\boldsymbol{x}) = \sum_{j \in \mathcal{J}} f_j(\boldsymbol{x}) \quad \text{and} \quad \partial_c f(\boldsymbol{x}) = \sum_{j \in \mathcal{J}} \partial_c f_j(\boldsymbol{x}). \tag{4}$$

Second, we can obtain the subdifferential of f of the form (2) where f_j involved are convex with

$$\partial_c f(\boldsymbol{x}) = \text{conv}\,\{\partial_c f_j(\boldsymbol{x}) \mid j \in \mathcal{J}(\boldsymbol{x})\}, \tag{5}$$

where conv denotes the convex hull of the set and $\mathcal{J}(\boldsymbol{x}) = \{j \in \mathcal{J} \mid f_j(\boldsymbol{x}) = f(\boldsymbol{x})\}$ is a set of active constraints.

The *generalized subdifferential* of a LLC function f at $\boldsymbol{x} \in \mathbb{R}^n$ is [6]

$$\partial f(\boldsymbol{x}) = \text{conv}\left\{\lim_{i \to \infty} \nabla f(\boldsymbol{x}_i) \mid \boldsymbol{x}_i \to \boldsymbol{x} \text{ and } \nabla f(\boldsymbol{x}_i) \text{ exists}\right\}.$$

If f is convex, then $\partial f(\boldsymbol{x})$ coincides with $\partial_c f(\boldsymbol{x})$.

The well-known necessary local optimality condition for unconstrained single-objective optimization with a LLC objective f at $\boldsymbol{x}^* \in \mathbb{R}^n$ is that $\boldsymbol{0} \in \partial f(\boldsymbol{x}^*)$. The point $\boldsymbol{x}^*$ is called *Clarke stationary* if this condition holds. Moreover, if f is convex, then the condition ensures global optimality. For the unconstrained single-objective DC problem with the objective $f = p - q$, if $\boldsymbol{x}^* \in \mathbb{R}^n$ is a local optimum for this problem, then $\partial q(\boldsymbol{x}^*) \subseteq \partial p(\boldsymbol{x}^*)$ [39]. Since this condition is hard to verify in practice, the relaxed condition

$$\partial p(\boldsymbol{x}^*) \cap \partial q(\boldsymbol{x}^*) \neq \emptyset \tag{6}$$

is often used [39] and $\boldsymbol{x}^*$ satisfying this is called a *critical point*. If $\boldsymbol{x}^*$ is Clarke stationary, then it is also critical.

Next we consider the concept of optimality in constrained multiobjective optimization. The solution $\boldsymbol{x}^* \in X$ is a *global Pareto optimum* for the problem (1) if there does not exist another solution $\boldsymbol{x} \in X$ such that $f_i(\boldsymbol{x}) \leq f_i(\boldsymbol{x}^*)$ for all $i \in I$ and $f_j(\boldsymbol{x}) < f_j(\boldsymbol{x}^*)$ for at least one $j \in I$. The solution $\boldsymbol{x}^* \in X$ is a *global weak Pareto optimum* for the problem (1), if there does not exist another solution $\boldsymbol{x} \in X$ such that $f_i(\boldsymbol{x}) < f_i(\boldsymbol{x}^*)$ for all $i \in I$. Moreover, $\boldsymbol{x}^* \in X$ is a *local (weak) Pareto optimum* if there exists $\delta > 0$ such that $\boldsymbol{x}^* \in X$ is a global (weak) Pareto optimum on $X \cap B(\boldsymbol{x}^*; \delta)$. Based on the above definitions, every Pareto optimum is a weak Pareto optimum.

In order to give an optimality condition for the constrained multiobjective problem, we define some concepts related to cones of the set $S \subseteq \mathbb{R}^n$ [2]. First, a set S is a *cone* if $\lambda \boldsymbol{x} \in S$ for all $\lambda \geq 0$ and $\boldsymbol{x} \in S$. We denote by $\operatorname{ray} S = \{\lambda \boldsymbol{s} \mid \lambda \geq 0, \boldsymbol{s} \in S\}$ and $\operatorname{cone} S = \operatorname{ray} \operatorname{conv} S$. Furthermore, we define a *contingent cone* at $\boldsymbol{x} \in S$ and a *polar cone*, respectively,

$$K_S(\boldsymbol{x}) = \{\boldsymbol{d} \in \mathbb{R}^n \mid \text{ there exist } t_i \downarrow 0 \text{ and } \boldsymbol{d}_i \to \boldsymbol{d} \text{ with } \boldsymbol{x} + t_i \boldsymbol{d}_i \in S\},$$
$$S^{\leq} = \{\boldsymbol{d} \in \mathbb{R}^n \mid \boldsymbol{s}^T \boldsymbol{d} \leq 0, \text{ for all } s \in S\}.$$

Throughout the paper, we denote by

$$F(\boldsymbol{x}) = \bigcup_{i=1}^{k} \partial f_i(\boldsymbol{x}) \text{ and } G(\boldsymbol{x}) = \bigcup_{l \in L(\boldsymbol{x})} \partial g_l(\boldsymbol{x}),$$

where $L(\boldsymbol{x}) = \{l \in L \mid g_l(\boldsymbol{x}) = 0\}$. In the following, we state a necessary condition for local weak Pareto optimality.

Theorem 2.1. *[24] If $\boldsymbol{x}^* \in X$ is a local weak Pareto optimum for the problem* (1)*, and the constraint qualification $G^{\leq}(\boldsymbol{x}) \subseteq K_X(\boldsymbol{x})$ holds, then*

$$\boldsymbol{0} \in \operatorname{conv} F(\boldsymbol{x}^*) + \operatorname{cl} \operatorname{cone} G(\boldsymbol{x}^*), \tag{7}$$

where cl *is a closure of the set.*

If the point $\boldsymbol{x}^*$ satisfies the condition (7), then it is called *weakly Pareto stationary*.

3 Multiobjective double bundle method for DC functions

In this section, we describe with details the new multiobjective double bundle method (MDBDC) solving multiobjective optimization problems (1) with DC functions as objectives and constraints. The method is descent, since it improves all the objectives simultaneously at every iteration.

The basic idea of MDBDC is to use the same framework as the multiobjective proximal bundle method (MPB) [25, 28]. We use the strategy for handling several objectives and constraints basing on the techniques in [18, 25, 40]. With this strategy, we transform the multiobjective problem to a single-objective one. After that, we employ a new cutting plane model similar to the one used in the proximal bundle method for DC functions (PBDC) [15]. This model uses explicitly the DC decomposition of the new objective in order to capture both the convex and the concave behaviour of it. Finally, we modify the double bundle method for DC functions (DBDC) [16] to solve the single-objective problem and to obtain the weakly Pareto stationary solution for the original multiobjective problem (1).

3.1 Cutting plane model for DC functions and direction finding

We define the *improvement function* $H : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ [18, 40] by

$$H(\boldsymbol{x}, \boldsymbol{y}) = \max\{f_i(\boldsymbol{x}) - f_i(\boldsymbol{y}),\ g_l(\boldsymbol{x}) \,|\, i \in I, l \in L\}. \tag{8}$$

Since $H(\,\cdot\,, \boldsymbol{y})$ is a maximum of DC functions, it is a DC function and its DC decomposition can be obtained as in [11]. Let the DC decompositions of f_i and g_l be $f_i = p_i - q_i$ for all $i \in I$ and $g_l = r_l - s_l$ for all $l \in L$. The functions f_i and g_l can be rewritten as

$$\begin{aligned} f_i(\boldsymbol{x}) &= p_i(\boldsymbol{x}) + \sum_{\substack{j=1 \\ j\neq i}}^{k} q_j(\boldsymbol{x}) + \sum_{l=1}^{m} s_l(\boldsymbol{x}) - \sum_{j=1}^{k} q_j(\boldsymbol{x}) - \sum_{l=1}^{m} s_l(\boldsymbol{x}) \\ g_l(\boldsymbol{x}) &= r_l(\boldsymbol{x}) + \sum_{\substack{j=1 \\ j\neq l}}^{m} s_j(\boldsymbol{x}) + \sum_{i=1}^{k} q_i(\boldsymbol{x}) - \sum_{i=1}^{k} q_i(\boldsymbol{x}) - \sum_{j=1}^{m} s_j(\boldsymbol{x}). \end{aligned}$$

In order to simplify the notations, we denote

$$A_i(\boldsymbol{x}, \boldsymbol{y}) = p_i(\boldsymbol{x}) + \sum_{\substack{j=1 \\ j\neq i}}^{k} q_j(\boldsymbol{x}) + \sum_{l=1}^{m} s_l(\boldsymbol{x}) - f_i(\boldsymbol{y}) \quad \text{and} \tag{9}$$

$$B_l(\boldsymbol{x}) = r_l(\boldsymbol{x}) + \sum_{\substack{j=1 \\ j\neq l}}^{m} s_j(\boldsymbol{x}) + \sum_{i=1}^{k} q_i(\boldsymbol{x}).$$

Now the DC decomposition of $H(\,\cdot\,, \boldsymbol{y})$ can be written as $H(\boldsymbol{x}, \boldsymbol{y}) = H_1(\boldsymbol{x}, \boldsymbol{y}) - H_2(\boldsymbol{x})$, where

$$H_1(\boldsymbol{x}, \boldsymbol{y}) = \max\{A_i(\boldsymbol{x}, \boldsymbol{y}), B_l(\boldsymbol{x}) \,|\, i \in I, l \in L\} \quad \text{and} \tag{10}$$

$$H_2(\boldsymbol{x}) = \sum_{i=1}^{k} q_i(\boldsymbol{x}) + \sum_{l=1}^{m} s_l(\boldsymbol{x})$$

and both $H_1(\,\cdot\,, \boldsymbol{y})$ and H_2 are convex respect to $\boldsymbol{x}$. Throughout the paper, the vector $\boldsymbol{y}$ in (8)–(10) is treated as a constant. Therefore, for instance, $\partial H(\boldsymbol{x}, \boldsymbol{y})$ is calculated respect to $\boldsymbol{x}$.

The improvement function $H(\,\cdot\,, \boldsymbol{y})$ has the following elementary properties [25, 40] justifying the name and the use of it.

Theorem 3.1. *The improvement function* $H(\,\cdot\,, \boldsymbol{y})$ (8) *has the following properties:*

(i) If $H(\boldsymbol{x}, \boldsymbol{y}) < H(\boldsymbol{y}, \boldsymbol{y})$, $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^n$, *then* $f_i(\boldsymbol{x}) < f_i(\boldsymbol{y})$ *for all* $i \in I$ *and* $g_l(\boldsymbol{x}) < 0$ *for all* $l \in L$.

(ii) If the solution $\boldsymbol{x}^* \in X$ *is a global weak Pareto optimum of the problem* (1), *then*

$$\boldsymbol{x}^* = \arg\min_{\boldsymbol{x} \in \mathbb{R}^n} H(\boldsymbol{x}, \boldsymbol{x}^*).$$

(iii) If $\mathbf{0} \in \partial H(\boldsymbol{x}^*, \boldsymbol{x}^*)$, *then the solution* $\boldsymbol{x}^* \in X$ *of the problem* (1) *is weakly Pareto stationary.*

Proof. (i) The claim follows immediately from the definition of $H(\,\cdot\,, \boldsymbol{y})$ (8).
(ii) Assume that the solution $\boldsymbol{x}^*$ is a global weak Pareto optimum of the problem (1). Thus, $\boldsymbol{x}^* \in X$ implying that $g_l(\boldsymbol{x}^*) \leq 0$ for all $l \in L$. Hence, $H(\boldsymbol{x}^*, \boldsymbol{x}^*) = 0$.

Suppose, contrary to our claim, that $\boldsymbol{x}^* \neq \arg\min_{\boldsymbol{x} \in \mathbb{R}^n} H(\boldsymbol{x}, \boldsymbol{x}^*)$. Now there exists $\boldsymbol{y}^* \in \mathbb{R}^n$ such that $H(\boldsymbol{y}^*, \boldsymbol{x}^*) < H(\boldsymbol{x}^*, \boldsymbol{x}^*) = 0$. Based on (i), we know that $f_i(\boldsymbol{y}^*) < f_i(\boldsymbol{x}^*)$ for all $i \in I$ and $g_l(\boldsymbol{y}^*) < 0$ for all $l \in L$ meaning that $\boldsymbol{y}^* \in X$. But this contradicts the assumption that $\boldsymbol{x}^*$ is a globally weakly Pareto optimal solution for the problem (1).
(iii) According to Theorem 3.23 in [2] and Lemma 2.10 in [24], we obtain

$$\begin{aligned}\mathbf{0} \in \partial H(\boldsymbol{x}^*, \boldsymbol{x}^*) &= \text{conv}\,\big\{F(\boldsymbol{x}^*) \,\cup\, G(\boldsymbol{x}^*)\big\} \\ &\subseteq \text{conv}\,\big\{\text{conv}\,F(\boldsymbol{x}^*) \,\cup\, \text{conv}\,G(\boldsymbol{x}^*)\big\} \\ &= \big\{\lambda\,\text{conv}\,F(\boldsymbol{x}^*) + (1-\lambda)\,\text{conv}\,G(\boldsymbol{x}^*) \mid \lambda \in [0,1]\big\}.\end{aligned}$$

Thus there exists $\lambda^* \in [0,1]$ such that $\mathbf{0} \in \text{conv}\,F(\boldsymbol{x}^*) + \text{cl cone}\,G(\boldsymbol{x}^*)$. Indeed, if $\lambda^* \in (0,1]$, we observe

$$\begin{aligned}\mathbf{0} &\in \text{conv}\,F(\boldsymbol{x}^*) + \frac{1-\lambda^*}{\lambda^*}\text{conv}\,G(\boldsymbol{x}^*) \\ &\subseteq \text{conv}\,F(\boldsymbol{x}^*) + \text{ray conv}\,G(\boldsymbol{x}^*) \\ &= \text{conv}\,F(\boldsymbol{x}^*) + \text{cone}\,G(\boldsymbol{x}^*) \\ &\subseteq \text{conv}\,F(\boldsymbol{x}^*) + \text{cl cone}\,G(\boldsymbol{x}^*).\end{aligned}$$

On the other hand, if $\lambda^* = 0$, then

$$\mathbf{0} \subseteq \text{conv}\,G(\boldsymbol{x}^*) \subseteq \text{cone}\,G(\boldsymbol{x}^*) \subseteq \text{conv}\,F(\boldsymbol{x}^*) + \text{cl cone}\,G(\boldsymbol{x}^*).$$

Therefore, $\boldsymbol{x}^*$ satisfies the condition (7) implying weak Pareto stationarity of $\boldsymbol{x}^*$. □

In the following, the index h relates to the h-th iteration and the current iteration point is denoted by $\boldsymbol{x}_h \in \mathbb{R}^n$. We assume, that at each auxiliary point $\boldsymbol{y}_j \in \mathbb{R}^n$ from the previous iterations, we can evaluate $p_i(\boldsymbol{y}_j)$, $q_i(\boldsymbol{y}_j)$, $r_l(\boldsymbol{y}_j)$, and $s_l(\boldsymbol{y}_j)$ and arbitrary $\boldsymbol{\xi}_{p,i}(\boldsymbol{y}_j) \in \partial p_i(\boldsymbol{y}_j)$, $\boldsymbol{\xi}_{q,i}(\boldsymbol{y}_j) \in \partial q_i(\boldsymbol{y}_j)$, $\boldsymbol{\xi}_{r,l}(\boldsymbol{y}_j) \in \partial r_l(\boldsymbol{y}_j)$, and $\boldsymbol{\xi}_{s,l}(\boldsymbol{y}_j) \in \partial s_l(\boldsymbol{y}_j)$ for all $i \in I$ and $l \in L$. From these, $A_i(\boldsymbol{y}_j, \boldsymbol{x}_h)$, $B_l(\boldsymbol{y}_j)$, $H_1(\boldsymbol{y}_j, \boldsymbol{x}_h)$, and $H_2(\boldsymbol{y}_j)$ can be composed by using formulas (9)–(10). Due to the convexity, their subgradients $\boldsymbol{a}_i$, $\boldsymbol{b}_l$, $\boldsymbol{h}_1$, and $\boldsymbol{h}_2$, respectively, are obtained by using the subdifferential calculus rules (4) and (5).

We collect information from the previous iterations into separate bundles. The bundles corresponding to $A_i(\,\cdot\,, \boldsymbol{x}_h)$ and B_l are

$$\begin{aligned} \mathcal{B}^h_{A,i} &= \left\{(\boldsymbol{y}_j, A_i(\boldsymbol{y}_j, \boldsymbol{x}_h), \boldsymbol{a}_{i,j}) \mid j \in J_1^h\right\} \quad \text{and} \\ \mathcal{B}^h_{B,l} &= \left\{(\boldsymbol{y}_j, B_l(\boldsymbol{y}_j), \boldsymbol{b}_{l,j}) \mid j \in J_1^h\right\} \end{aligned} \tag{11}$$

for all $i \in I$, $l \in L$, where $\boldsymbol{a}_{i,j} \in \partial A_i(\boldsymbol{y}_j, \boldsymbol{x}_h)$ and $\boldsymbol{b}_{l,j} \in \partial B_l(\boldsymbol{y}_j)$ are the arbitrary subgradients and J_1^h is the set of indices. Moreover, we define bundles

$$\mathcal{B}^h_A = \bigcup_{i=1}^{k} \mathcal{B}^h_{A,i}, \quad \mathcal{B}^h_B = \bigcup_{l=1}^{m} \mathcal{B}^h_{B,l}, \quad \text{and} \quad \mathcal{B}^h_1 = \mathcal{B}^h_A \bigcup \mathcal{B}^h_B.$$

For every $j \in J_1^h$, we have one element in the bundle $\mathcal{B}_1^h$ corresponding to the triplet $(\boldsymbol{y}_j, H_1(\boldsymbol{y}_j, \boldsymbol{x}_h), \boldsymbol{h}_{1,j})$, where $\boldsymbol{h}_{1,j} \in \partial H_1(\boldsymbol{y}_j, \boldsymbol{x}_h)$. The bundle related to H_2 is

$$\mathcal{B}^h_2 = \left\{(\boldsymbol{y}_j, H_2(\boldsymbol{y}_j), \boldsymbol{h}_{2,j}) \mid j \in J_2^h\right\}, \tag{12}$$

where $\boldsymbol{h}_{2,j} \in \partial H_2(\boldsymbol{y}_j)$ and J_2^h is the set of indices. Note that the only restriction for the bundles $\mathcal{B}_1^h$ and $\mathcal{B}_2^h$ is that they must contain the triplets related to the current iteration point $\boldsymbol{x}_h$.

In the spirit of Theorem 3.1, we derive a method producing solutions $\boldsymbol{x}^* \in X$ such that $\boldsymbol{0} \in \partial H(\boldsymbol{x}^*, \boldsymbol{x}^*)$. First, our aim is to find a search direction $\boldsymbol{d}^h \in \mathbb{R}^n$ by solving the problem

$$\min_{\boldsymbol{d} \in \mathbb{R}^n} \quad H(\boldsymbol{x}_h + \boldsymbol{d}, \boldsymbol{x}_h). \tag{13}$$

In order to approximate the problem (13), we utilize the cutting plane model basing on the one presented in [15]. With this new model, we can take into account both the convex and the concave behaviour of $H(\,\cdot\,, \boldsymbol{x}_h)$ by linearizing its DC components separately.

The convex DC components of $H(\,\cdot\,, \boldsymbol{x}_h)$ can be linearized by using the classical cutting plane model [19, 26, 36]. We linearize all the components $A_i(\,\cdot\,, \boldsymbol{x}_h)$ and $B_l(\boldsymbol{x}_h)$ of the first DC component $H_1(\,\cdot\,, \boldsymbol{x}_h)$:

$$\begin{aligned} \hat{A}_i^h(\boldsymbol{x}) &= \max_{j \in J_1^h} \left\{A_i(\boldsymbol{x}_h, \boldsymbol{x}_h) + \boldsymbol{a}_{i,j}^T(\boldsymbol{x} - \boldsymbol{x}_h) - \alpha^A_{i,j}\right\} \text{ and} \\ \hat{B}_l^h(\boldsymbol{x}) &= \max_{j \in J_1^h} \left\{B_l(\boldsymbol{x}_h) + \boldsymbol{b}_{l,j}^T(\boldsymbol{x} - \boldsymbol{x}_h) - \alpha^B_{l,j}\right\}, \end{aligned}$$

where $\boldsymbol{a}_{i,j} \in \partial A_i(\boldsymbol{y}_j, \boldsymbol{x}_h)$ and $\boldsymbol{b}_{l,j} \in \partial B_l(\boldsymbol{y}_j)$ for $j \in J_1^h$. The linearization errors evaluated at $\boldsymbol{x}_h$ for all $j \in J_1^h$ are

$$\alpha_{i,j}^A = \alpha_i^A(\boldsymbol{x}_h, \boldsymbol{y}_j) = A_i(\boldsymbol{x}_h, \boldsymbol{x}_h) - A_i(\boldsymbol{y}_j, \boldsymbol{x}_h) - \boldsymbol{a}_{i,j}^T(\boldsymbol{x}_h - \boldsymbol{y}_j) \text{ for all } i \in I,$$
$$\alpha_{l,j}^B = \alpha_l^B(\boldsymbol{x}_h, \boldsymbol{y}_j) = B_l(\boldsymbol{x}_h) - B_l(\boldsymbol{y}_j) - \boldsymbol{b}_{l,j}^T(\boldsymbol{x}_h - \boldsymbol{y}_j) \text{ for all } l \in L.$$

Additionally, for each $j \in J_1^h$ we denote by $\alpha_{1,j}^H$ the linearization error associated with the triplet $(\boldsymbol{y}_j, H_1(\boldsymbol{y}_j, \boldsymbol{x}_h), \boldsymbol{h}_{1,j}) \in \mathcal{B}_1^h$. Thus, the linearization of the first DC component H_1 is

$$\hat{H}_1^h(\boldsymbol{x}) = \max\{\hat{A}_i^h(\boldsymbol{x}),\ \hat{B}_l^h(\boldsymbol{x}) \,|\, i \in I, l \in L\}. \tag{14}$$

Similarly, we can linearize the second DC component H_2 by

$$\hat{H}_2^h(\boldsymbol{x}) = \max_{j \in J_2^h}\{H_2(\boldsymbol{x}_h) + \boldsymbol{h}_{2,j}^T(\boldsymbol{x} - \boldsymbol{x}_h) - \alpha_{2,j}^H\}, \tag{15}$$

where $\boldsymbol{h}_{2,j} \in \partial H_2(\boldsymbol{y}_j)$ for $j \in J_2^h$ and the linearization error evaluated at $\boldsymbol{x}_h$ for all $j \in J_2^h$ is

$$\alpha_{2,j}^H = \alpha_2^H(\boldsymbol{x}_h, \boldsymbol{y}_j) = H_2(\boldsymbol{x}_h) - H_2(\boldsymbol{y}_j) - \boldsymbol{h}_{2,j}^T(\boldsymbol{x}_h - \boldsymbol{y}_j).$$

Furthermore, all the linearization errors are nonnegative [19].

Finally, we approximate $H(\,\cdot\,, \boldsymbol{x}_h)$ by combining the convex cutting plane models of its DC components. Thus, we obtain the following piecewise linear, nonconvex DC approximation of $H(\,\cdot\,, \boldsymbol{x}_h)$:

$$\hat{H}^h(\boldsymbol{x}) = \hat{H}_1^h(\boldsymbol{x}) - \hat{H}_2^h(\boldsymbol{x}).$$

The problem (13) is now estimated with the nonsmooth, nonconvex, and quadratic DC problem

$$\min_{\boldsymbol{d} \in \mathbb{R}^n} \quad P^h(\boldsymbol{d}) = \hat{H}_1^h(\boldsymbol{x}_h + \boldsymbol{d}) - \hat{H}_2^h(\boldsymbol{x}_h + \boldsymbol{d}) + \frac{1}{2t}\|\boldsymbol{d}\|^2, \tag{16}$$

where $t > 0$ is a proximity parameter used widely at bundle methods while the convex stabilizing term $\frac{1}{2t}\|\boldsymbol{d}\|^2$ keeps the approximation local enough and ensures the existence of the search direction. The search direction obtained as a solution of the problem (16) is denoted by $\boldsymbol{d}_t^h$.

We give the following properties making it legitimate to apply the new cutting plane model $\hat{H}^h$.

Lemma 3.2. *The following properties hold:*

(i) $\hat{H}_1^h(\boldsymbol{x}_h + \boldsymbol{d}) \le H_1(\boldsymbol{x}_h + \boldsymbol{d}, \boldsymbol{x}_h)$ *and* $\hat{H}_2^h(\boldsymbol{x}_h + \boldsymbol{d}) \le H_2(\boldsymbol{x}_h + \boldsymbol{d})$.

(ii) For any $t > 0$, *we have* $\hat{H}^h(\boldsymbol{x}_h + \boldsymbol{d}_t^h) - H(\boldsymbol{x}_h, \boldsymbol{x}_h) \le -\frac{1}{2t}\|\boldsymbol{d}_t^h\|^2 \le 0$.

Proof. (i) These follow immediately from the definition of the cutting plane model. (ii) For the feasible solution $\boldsymbol{d}' = \mathbf{0}$ of the problem (16), $\hat{H}_1^h(\boldsymbol{x}_h + \mathbf{0}) \leq H_1(\boldsymbol{x}_h, \boldsymbol{x}_h)$ and

$$\hat{H}_2^h(\boldsymbol{x}_h + \mathbf{0}) = \max_{j \in J_2^h}\{H_2(\boldsymbol{x}_h) - \alpha_{2,j}^H\} = H_2(\boldsymbol{x}_h),$$

since $\boldsymbol{x}_h$ is included in $\mathcal{B}_2^h$ implying that there exists at least one $j \in J_2^h$ such that $\alpha_{2,j}^H = 0$. Thus,

$$\begin{aligned}\hat{H}^h(\boldsymbol{x}_h + \boldsymbol{d}') + \frac{1}{2t}\|\boldsymbol{d}'\|^2 &= \hat{H}_1^h(\boldsymbol{x}_h + \mathbf{0}) - \hat{H}_2^h(\boldsymbol{x}_h + \mathbf{0}) \\ &\leq H_1(\boldsymbol{x}_h, \boldsymbol{x}_h) - H_2(\boldsymbol{x}_h) = H(\boldsymbol{x}_h, \boldsymbol{x}_h),\end{aligned}$$

and for the global solution $\boldsymbol{d}_t^h$ of the problem (16), we obtain

$$\hat{H}^h(\boldsymbol{x}_h + \boldsymbol{d}_t^h) + \frac{1}{2t}\|\boldsymbol{d}_t^h\|^2 \leq \hat{H}^h(\boldsymbol{x}_h + \boldsymbol{d}') + \frac{1}{2t}\|\boldsymbol{d}'\|^2 \leq H(\boldsymbol{x}_h, \boldsymbol{x}_h).$$

□

The solution $\boldsymbol{d}_t^h$ of the problem (16) can be shown to be always bounded.

Lemma 3.3. *For any proximity parameter $t > 0$, it holds that*

$$\|\boldsymbol{d}_t^h\| \leq 2t\big(\|\boldsymbol{h}_1(\boldsymbol{x}_h)\| + \|\boldsymbol{h}_{2,\max}\|\big),$$

where $\boldsymbol{h}_1(\boldsymbol{x}_h) \in \partial H_1(\boldsymbol{x}_h, \boldsymbol{x}_h)$ and $\|\boldsymbol{h}_{2,\max}\| = \max_{j \in J_2^h}\{\|\boldsymbol{h}_{2,j}\|\}$.

Proof. Our proof begins with the following observation basing on the definition of $\hat{H}_2^h$ (15):

$$\hat{H}_2^h(\boldsymbol{x}_h + \boldsymbol{d}) \leq H_2(\boldsymbol{x}_h) + \max_{j \in J_2^h}\left\{\boldsymbol{h}_{2,j}^T \boldsymbol{d}\right\} \leq H_2(\boldsymbol{x}_h) + \|\boldsymbol{h}_{2,\max}\|\|\boldsymbol{d}\|. \tag{17}$$

Now the element $(\boldsymbol{x}_h, H_1(\boldsymbol{x}_h, \boldsymbol{x}_h), \boldsymbol{h}_1(\boldsymbol{x}_h))$, where $\boldsymbol{h}_1(\boldsymbol{x}_h) \in \partial H_1(\boldsymbol{x}_h, \boldsymbol{x}_h)$, belongs to the bundle $\mathcal{B}_1^h$, and from the definition of $\hat{H}_1^h$ (14), it follows that for all $\boldsymbol{d} \in \mathbb{R}^n$

$$\begin{aligned}\hat{H}_1^h(\boldsymbol{x}_h + \boldsymbol{d}) &\geq H_1(\boldsymbol{x}_h, \boldsymbol{x}_h) + \boldsymbol{h}_1(\boldsymbol{x}_h)^T \boldsymbol{d} - \alpha_1^H \\ &= H_1(\boldsymbol{x}_h, \boldsymbol{x}_h) + \boldsymbol{h}_1(\boldsymbol{x}_h)^T \boldsymbol{d},\end{aligned} \tag{18}$$

where α_1^H is the linearization error associated to $(\boldsymbol{x}_h, H_1(\boldsymbol{x}_h, \boldsymbol{x}_h), \boldsymbol{h}_1(\boldsymbol{x}_h))$ and it is equal to zero.

By combining (17) and (18) we obtain

$$\begin{aligned}\hat{H}^h(\boldsymbol{x}_h + \boldsymbol{d}) &= \hat{H}_1^h(\boldsymbol{x}_h + \boldsymbol{d}) - \hat{H}_2^h(\boldsymbol{x}_h + \boldsymbol{d}) \\ &\geq H_1(\boldsymbol{x}_h, \boldsymbol{x}_h) - H_2(\boldsymbol{x}_h) + \boldsymbol{h}_1(\boldsymbol{x}_h)^T \boldsymbol{d} - \|\boldsymbol{h}_{2,\max}\|\|\boldsymbol{d}\| \\ &\geq H(\boldsymbol{x}_h, \boldsymbol{x}_h) - \Big(\|\boldsymbol{h}_1(\boldsymbol{x}_h)\| + \|\boldsymbol{h}_{2,\max}\|\Big)\|\boldsymbol{d}\|\end{aligned}$$

for all $\boldsymbol{d} \in \mathbb{R}^n$. According to Lemma 3.2 (ii),

$$-\frac{1}{2t}\|\boldsymbol{d}_t^h\|^2 \geq \hat{H}^h(\boldsymbol{x}_h + \boldsymbol{d}_t^h) - H(\boldsymbol{x}_h, \boldsymbol{x}_h) \geq -\Big(\|\boldsymbol{h}_1(\boldsymbol{x}_h)\| + \|\boldsymbol{h}_{2,\max}\|\Big)\|\boldsymbol{d}_t^h\|$$

implying the statement. □

In order to solve globally the direction finding problem (16), we notice that the DC components of P^h are $\hat{H}_1^h(\boldsymbol{x}_h + \boldsymbol{d}) + \frac{1}{2t}\|\boldsymbol{d}\|^2$ and $\hat{H}_2^h(\boldsymbol{x}_h + \boldsymbol{d})$. Furthermore, the second DC component $\hat{H}_2^h$ is polyhedral convex meaning that $\hat{H}_2^h$ is of the form $\max\{\boldsymbol{u}_k^T\boldsymbol{x} - v_k \,|\, k \in \mathcal{K}\}$, where $\boldsymbol{u}_k \in \mathbb{R}^n$, $v_k \in \mathbb{R}$, and $\mathcal{K}$ is finite. Thus, we employ the solution approach presented in [20, 21, 33] to obtain the global solution of the problem (16).

We can reformulate the objective function P^h of the problem (16) by recalling (15) in the form

$$P^h(\boldsymbol{d}) = \min_{j \in J_2^h} \big\{P_j^h(\boldsymbol{d}) = \hat{H}_1^h(\boldsymbol{x}_h + \boldsymbol{d}) - H_2(\boldsymbol{x}_h) - \boldsymbol{h}_{2,j}^T\boldsymbol{d} + \alpha_{2,j}^H + \frac{1}{2t}\|\boldsymbol{d}\|^2\big\}.$$

Therefore, we obtain

$$\min_{\boldsymbol{d} \in \mathbb{R}^n} \min_{j \in J_2^h} \big\{P_j^h(\boldsymbol{d})\big\} = \min_{j \in J_2^h} \min_{\boldsymbol{d} \in \mathbb{R}^n} \big\{P_j^h(\boldsymbol{d})\big\},$$

and for this reason, the order of the minimization can be changed in the problem (16). Thus, due to the size of the bundle $\mathcal{B}_2^h$, we solve $|J_2^h|$ convex, nonsmooth subproblems for $j \in J_2^h$

$$\min_{\boldsymbol{d} \in \mathbb{R}^n} P_j^h(\boldsymbol{d}) = \hat{H}_1^h(\boldsymbol{x}_h + \boldsymbol{d}) - H_2(\boldsymbol{x}_h) - \boldsymbol{h}_{2,j}^T\boldsymbol{d} + \alpha_{2,j}^H + \frac{1}{2t}\|\boldsymbol{d}\|^2, \tag{19}$$

and the solution of the subproblem $j \in J_2^h$ is denoted by $\boldsymbol{d}_t^h(j)$. Moreover, the overall global solution $\boldsymbol{d}_t^h$ of the problem (16) is $\boldsymbol{d}_t^h = \boldsymbol{d}_t^h(j^*)$, where the index $j^* = \arg\min \big\{P_j^h\big(\boldsymbol{d}_t^h(j)\big) \,|\, j \in J_2^h\big\}$. In practice, the size of $\mathcal{B}_2^h$ can be freely chosen such that $|J_2^h| \geq 1$, and therefore we can control the amount of computation. The solution process can be eased by rewriting (19) as a smooth problem and solving its dual.

3.2 Guaranteeing weak Pareto stationarity

Many methods for single-objective DC optimization with the objective $f = p - q$ stop after finding a critical point $\boldsymbol{x}' \in \mathbb{R}^n$ satisfying (6), since the stronger necessary optimality condition $\partial q(\boldsymbol{x}^*) \subseteq \partial p(\boldsymbol{x}^*)$ for a local optimum $\boldsymbol{x}^* \in \mathbb{R}^n$ is hard to verify in practice. Reason for this is that we usually do not know, or cannot calculate, the whole subdifferentials of DC components. Whenever $\boldsymbol{x}'$ is critical, then $\boldsymbol{0} \in \partial p(\boldsymbol{x}') - \partial q(\boldsymbol{x}')$. However, the subdifferential calculus [6] only implies

$$\partial f(\boldsymbol{x}') \subseteq \partial p(\boldsymbol{x}') - \partial q(\boldsymbol{x}'), \tag{20}$$

where the equality holds if either p or q is differentiable at $\boldsymbol{x}'$. Hence, we might end up with a solution such that $\mathbf{0} \in \partial p(\boldsymbol{x}') - \partial q(\boldsymbol{x}')$ but $\mathbf{0} \notin \partial f(\boldsymbol{x}')$. Due to this, $\boldsymbol{x}'$ might not be a local optimum or even a saddle point. Therefore, criticality is a weaker condition than Clarke stationarity $\mathbf{0} \in \partial f(\boldsymbol{x}^*)$ for a local optimum $\boldsymbol{x}^* \in \mathbb{R}^n$, which is often obtained in nonconvex optimization.

This kind of observation can be made in multiobjective optimization as well by comparing weak Pareto stationarity (7) with the multiobjective Pareto criticality condition given in [14]: in the unconstrained case, the solution $\boldsymbol{x}' \in \mathbb{R}^n$ is called *Pareto critical* if

$$\mathbf{0} \in \operatorname{conv} \{\partial p_i(\boldsymbol{x}') - \partial q_i(\boldsymbol{x}') \mid i \in I\}. \tag{21}$$

Indeed, it is easy to see that a weakly Pareto stationary point $\boldsymbol{x}^*$ satisfies

$$\mathbf{0} \in \operatorname{conv} \{\partial f_i(\boldsymbol{x}^*) \mid i \in I\} \subseteq \operatorname{conv} \{\partial p_i(\boldsymbol{x}^*) - \partial q_i(\boldsymbol{x}^*) \mid i \in I\},$$

by applying (20) to (7). Thus, weak Pareto stationarity implies Pareto criticality.

However, the inverse does not necessarily hold. Consider the unconstrained case of the problem (1) with two objectives having DC components as follows:

$$\begin{aligned} p_1(x) &= \max\{-x, 2x\}, \quad q_1(x) = \max\{-2x, x\}, \\ p_2(x) &= \max\{x^2, x\}, \quad q_2(x) = \max\{0.5x^2, -x\}, \end{aligned}$$

where $x \in \mathbb{R}$. In order to verify the condition (21), we consider the point $x' = 0$ and check whether the intersection

$$\lambda \partial p_1(x') + (1-\lambda)\partial p_2(x') \, \cap \, \lambda \partial q_1(x') + (1-\lambda)\partial q_2(x')$$

is an empty set or not. With $\lambda = 1$, the intersection is $[-1, 2] \cap [-2, 1] = [-1, 1]$. This set is nonempty, and thus, the point x' is Pareto critical. On the other hand, $0 \notin \operatorname{conv} \{\partial f_1(x'), \partial f_2(x')\} = \{1\}$, and thus x' is not weakly Pareto stationary.

In order to avoid the bad behaviour of Pareto critical points, we ensure weak Pareto stationarity in MDBDC. By Theorem 3.1 (iii), a point $\boldsymbol{x}^* \in \mathbb{R}^n$ is weakly Pareto stationary, if $\mathbf{0} \in \partial H(\boldsymbol{x}^*, \boldsymbol{x}^*)$, or in other words, if $\boldsymbol{x}^*$ is Clarke stationary for $H(\,\cdot\,, \boldsymbol{x}^*)$. Thus, it is sufficient to consider the single-objective DC problem (13). Since we utilize only the DC structure of this problem, a natural approach would be to verify the criticality condition (6). However, Clarke stationarity is harder to obtain, and to achieve this, we apply the stopping procedure presented in [16]. The beauty of this procedure lies in its ability to ensure that $\boldsymbol{\xi}_1 - \boldsymbol{\xi}_2$ such that $\boldsymbol{\xi}_1 \in \partial p(\boldsymbol{x})$, $\boldsymbol{\xi}_2 \in \partial q(\boldsymbol{x})$ belongs to the subdifferential of $f = p - q$ at $\boldsymbol{x} \in \mathbb{R}^n$. Moreover, if a point is not Clarke stationary, then the procedure generates a descent direction.

We describe here only the most essential parts of this procedure, and all the results presented regarding this procedure are valid for any DC function even if we give them here for $H(\,\cdot\,, \boldsymbol{y}) = H_1(\,\cdot\,, \boldsymbol{y}) - H_2$. For more details we refer to [16]. The stopping procedure needs one mild assumption holding in nearly all practical applications:

$\mathbf{A1}$: The subdifferentials $\partial H_1(\boldsymbol{x},\boldsymbol{y})$ and $\partial H_2(\boldsymbol{x})$ are polytopes for each $\boldsymbol{x} \in \mathbb{R}^n$.

Recall that the directional derivative of a convex function can be written like (3). Now we denote the directional derivatives of $H_1(\boldsymbol{x},\boldsymbol{y})$ and $H_2(\boldsymbol{x})$ respect to $\boldsymbol{x} \in \mathbb{R}^n$ in the direction $\boldsymbol{d} \in \mathbb{R}^n$ by

$$H_1'(\boldsymbol{x};\boldsymbol{d}) = \max\{\boldsymbol{w}^T\boldsymbol{d} \,|\, \boldsymbol{w} \in \partial H_1(\boldsymbol{x},\boldsymbol{y})\} \quad \text{and}$$
$$H_2'(\boldsymbol{x};\boldsymbol{d}) = \max\{\boldsymbol{w}^T\boldsymbol{d} \,|\, \boldsymbol{w} \in \partial H_2(\boldsymbol{x})\}.$$

For any $\boldsymbol{d} \in \mathbb{R}^n$, $\boldsymbol{d} \neq \boldsymbol{0}$, we define the sets

$$\sigma_1(\boldsymbol{x},\boldsymbol{y};\boldsymbol{d}) = \{\boldsymbol{\xi} \in \partial H_1(\boldsymbol{x},\boldsymbol{y}) \,|\, \boldsymbol{\xi}^T\boldsymbol{d} = H_1'(\boldsymbol{x};\boldsymbol{d})\} \quad \text{and}$$
$$\sigma_2(\boldsymbol{x};\boldsymbol{d}) = \{\boldsymbol{\xi} \in \partial H_2(\boldsymbol{x}) \,|\, \boldsymbol{\xi}^T\boldsymbol{d} = H_2'(\boldsymbol{x};\boldsymbol{d})\}.$$

Furthermore, let T_{DC} be a set of full measure at the point $\boldsymbol{x} \in \mathbb{R}^n$ such that $\sigma_1(\boldsymbol{x},\boldsymbol{y};\boldsymbol{d})$ and $\sigma_2(\boldsymbol{x};\boldsymbol{d})$ are singletons for any $\boldsymbol{d} \in T_{DC}$.

Theorem 3.4. *[16] Let $\boldsymbol{x},\boldsymbol{y} \in \mathbb{R}^n$, $\boldsymbol{d} \in T_{DC}$, $\sigma_1(\boldsymbol{x},\boldsymbol{y};\boldsymbol{d}) = \{\boldsymbol{\xi}_1\}$ and $\sigma_2(\boldsymbol{x};\boldsymbol{d}) = \{\boldsymbol{\xi}_2\}$. Then $\boldsymbol{\xi}_1 - \boldsymbol{\xi}_2 \in \partial H(\boldsymbol{x},\boldsymbol{y})$*

Based on this result, in order to compute $\boldsymbol{\xi} \in \partial H(\boldsymbol{x},\boldsymbol{y})$ utilizing the DC components, we need to find for any $\boldsymbol{d} \in \mathbb{R}^n$ a direction $\bar{\boldsymbol{d}} \in T_{DC}$ such that $\|\boldsymbol{d} - \bar{\boldsymbol{d}}\| < \delta$ for any sufficiently small $\delta \in (0,1)$.

The stopping procedure guaranteeing Clarke stationarity bases on the following result:

Theorem 3.5. *[16] Let $\boldsymbol{x} \in \mathbb{R}^n$, $\boldsymbol{d} \in \mathbb{R}^n$ be any direction such that $\boldsymbol{d} \neq \boldsymbol{0}$ and the assumption $\mathbf{A1}$ be valid. Then for a given $\boldsymbol{v} \in V$, where $V = \{\boldsymbol{v} \in \mathbb{R}^n \,|\, \boldsymbol{v} = (v_1,\ldots,v_n),\ |v_i| = 1,\ i = 1,\ldots,n\}$, there exists $\alpha_0 \in (0,1]$ such that for all $\alpha \in (0,\alpha_0]$:*

- *(i) $\bar{\boldsymbol{d}}(\alpha) = \boldsymbol{d} + \mathrm{e}^n(\alpha) \in T_{DC}$, where $\mathrm{e}^n(\alpha) = (\alpha v_1, \alpha^2 v_2, \ldots, \alpha^n v_n)$.*
- *(ii) $\sigma_1(\boldsymbol{x},\boldsymbol{y};\bar{\boldsymbol{d}}(\alpha)) \subset \sigma_1(\boldsymbol{x},\boldsymbol{y};\boldsymbol{d}) \subseteq \partial H_1(\boldsymbol{x},\boldsymbol{y})$ and $\sigma_2(\boldsymbol{x};\bar{\boldsymbol{d}}(\alpha)) \subset \sigma_2(\boldsymbol{x};\boldsymbol{d}) \subseteq \partial H_2(\boldsymbol{x})$ for all $\boldsymbol{y} \in \mathbb{R}^n$.*
- *(iii) $\boldsymbol{\xi}_1 - \boldsymbol{\xi}_2 \in \partial H(\boldsymbol{x},\boldsymbol{y})$ for $\boldsymbol{\xi}_1 \in \sigma_1(\boldsymbol{x},\boldsymbol{y};\bar{\boldsymbol{d}}(\alpha))$, $\boldsymbol{\xi}_2 \in \sigma_2(\boldsymbol{x};\bar{\boldsymbol{d}}(\alpha))$, and all $\boldsymbol{y} \in \mathbb{R}^n$.*

In order to estimate the subdifferential $\partial H(\boldsymbol{x},\boldsymbol{y})$, we briefly introduce the *Goldstein ε-subdifferential* for the improvement function H [26]:

$$\partial_\varepsilon^G H(\boldsymbol{x},\boldsymbol{y}) = \text{cl conv}\{\partial H(\boldsymbol{z},\boldsymbol{y}) \,|\, \boldsymbol{z} \in B(\boldsymbol{x};\varepsilon)\}.$$

Thus $\partial H(\boldsymbol{x},\boldsymbol{y}) \subseteq \partial_\varepsilon^G H(\boldsymbol{x},\boldsymbol{y})$ for each $\varepsilon \geq 0$ and the smaller the parameter ε is, the better estimate we get. In Algorithm 1, $D_1 = \{\boldsymbol{d} \in \mathbb{R}^n \,|\, \|\boldsymbol{d}\| = 1\}$ is a unit sphere in $\mathbb{R}^n$ and the set U_j is an approximation of $\partial_\varepsilon^G H(\boldsymbol{x},\boldsymbol{y})$ such that $U_j \subset \partial_\varepsilon^G H(\boldsymbol{x},\boldsymbol{y})$ for all iterations $j \geq 0$.

Algorithm 1 Guaranteeing Clarke stationarity

Data: The point $\boldsymbol{x} \in \mathbb{R}^n$ under consideration, the descent parameter $m_1 \in (0, 1)$, the stopping tolerance $\delta \in (0, 1)$, and the proximity measure $\varepsilon > 0$.

Step 0. *(Initialization)* Select the direction $\boldsymbol{d}_0 \in D_1$ and find $\bar{\boldsymbol{d}}_0(\alpha) \in T_{DC}$ at $\boldsymbol{x}$ using $\boldsymbol{d}_0$. Compute $\boldsymbol{\xi}_1 \in \sigma_1(\boldsymbol{x}, \boldsymbol{x}; \bar{\boldsymbol{d}}_0(\alpha))$ and $\boldsymbol{\xi}_2 \in \sigma_2(\boldsymbol{x}; \bar{\boldsymbol{d}}_0(\alpha))$. Set $U_0 = \{\boldsymbol{\xi}_1 - \boldsymbol{\xi}_2\}$, $\tilde{\boldsymbol{x}} = \boldsymbol{x}$ and $j = 0$.

Step 1. *(Clarke stationarity)* Find $\bar{\boldsymbol{u}}_j$ as the solution of the problem

$$\min_{\boldsymbol{u} \in U_j} \frac{1}{2}\|\boldsymbol{u}\|^2.$$

If $\|\bar{\boldsymbol{u}}_j\| \leq \delta$, then Clarke stationarity is obtained and EXIT with $\boldsymbol{x}^+ = \boldsymbol{x}$.

Step 2. *(Search direction)* Compute the search direction $\boldsymbol{d}_{j+1} = -\bar{\boldsymbol{u}}_j / \|\bar{\boldsymbol{u}}_j\|$.

Step 3. *(New subgradient)* Find $\bar{\boldsymbol{d}}_{j+1}(\alpha) \in T_{DC}$ at $\tilde{\boldsymbol{x}}$ using $\boldsymbol{d}_{j+1}$. Compute $\boldsymbol{\xi}_1 \in \sigma_1(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{x}}; \bar{\boldsymbol{d}}_{j+1}(\alpha))$ and $\boldsymbol{\xi}_2 \in \sigma_2(\tilde{\boldsymbol{x}}; \bar{\boldsymbol{d}}_{j+1}(\alpha))$. Set $\bar{\boldsymbol{\xi}}_{j+1} = \boldsymbol{\xi}_1 - \boldsymbol{\xi}_2$. If $\boldsymbol{x} \neq \tilde{\boldsymbol{x}}$, then go to Step 5.

Step 4. *(Descent test)* If $(\bar{\boldsymbol{\xi}}_{j+1})^T \boldsymbol{d}_{j+1} \leq -m_1 \|\bar{\boldsymbol{u}}_j\|$, then go to Step 6.

Step 5. *(Update)* Set $U_{j+1} = \text{conv}\,\{U_j \bigcup \{\bar{\boldsymbol{\xi}}_{j+1}\}\}$, $\tilde{\boldsymbol{x}} = \boldsymbol{x}$ and $j = j + 1$. Go to Step 1.

Step 6. *(Step-length)* Calculate

$$\beta^* = \arg\max\{\beta \geq 0 \mid H(\boldsymbol{x} + \beta \boldsymbol{d}_{j+1}, \boldsymbol{x}) - H(\boldsymbol{x}, \boldsymbol{x}) < 0\}.$$

If $\beta^* \geq \varepsilon$, then $\boldsymbol{x}^+ = \boldsymbol{x} + \beta^* \boldsymbol{d}_{j+1}$, and EXIT. Otherwise, $\tilde{\boldsymbol{x}} = \boldsymbol{x} + \beta^* \boldsymbol{d}_{j+1}$ and go to Step 3.

3.3 Algorithm

In this section, we combine the above presented subsections and give a detailed description of MDBDC. In order to guarantee the convergence of MDBDC, we suppose that $\mathbf{A1}$ is satisfied along with the following assumption for the level set at the starting point $\boldsymbol{x}_0 \in X$:

$\mathbf{A2}$: The level set $\mathcal{F}_0 = \{\boldsymbol{x} \in X \mid f_i(\boldsymbol{x}) \leq f_i(\boldsymbol{x}_0),$ for all $i \in I\}$ is compact.

To simplify the presentation, we divide MDBDC into four algorithms. Algorithm 2 gives the outline of the whole method while Algorithm 3 describes the main iteration of MDBDC being the heart of the method by producing new iteration points. Additionally, we use Algorithm 1 presented in Section 3.2 to ensure weak Pareto stationarity and the scaling procedure Algorithm 4 described later in this section. The scaling procedure is applied in order to avoid numerical difficulties.

The proximity parameter t is updated in two places: during the execution of the main iteration and between two main iterations. In the latter case, the updating procedure in Step 2 of Algorithm 2 is inspired by the weighting update method given in [19], and t may either increase or decrease. In Step 5 of Algorithm 3, t may only decrease.

We begin by discussing about Algorithm 2. First we notice that the linearization errors can be updated by using the following formulas for all $i \in I$ and $l \in L$

$$\begin{aligned} \alpha_i^A(\boldsymbol{x}_{h+1}, \boldsymbol{y}_j) =& \alpha_i^A(\boldsymbol{x}_h, \boldsymbol{y}_j) + \tilde{A}_i(\boldsymbol{x}_{h+1}) - \tilde{A}_i(\boldsymbol{x}_h) - \boldsymbol{a}_{i,j}^T(\boldsymbol{x}_{h+1} - \boldsymbol{x}_h) \\ \alpha_l^B(\boldsymbol{x}_{h+1}, \boldsymbol{y}_j) =& \alpha_l^B(\boldsymbol{x}_h, \boldsymbol{y}_j) + B_l(\boldsymbol{x}_{h+1}) - B_l(\boldsymbol{x}_h) - \boldsymbol{b}_{l,j}^T(\boldsymbol{x}_{h+1} - \boldsymbol{x}_h) \\ \alpha_2^H(\boldsymbol{x}_{h+1}, \boldsymbol{y}_j) =& \alpha_2^H(\boldsymbol{x}_h, \boldsymbol{y}_j) + H_2(\boldsymbol{x}_{h+1}) - H_2(\boldsymbol{x}_h) - \boldsymbol{h}_{2,j}^T(\boldsymbol{x}_{h+1} - \boldsymbol{x}_h), \end{aligned} \tag{22}$$

where $\tilde{A}_i(\boldsymbol{x}) = A(\boldsymbol{x}, \boldsymbol{x}) + f_i(\boldsymbol{x})$. Thus, we store only elements $(\boldsymbol{\xi}, \alpha)$ in $\mathcal{B}_1^h$ and $\mathcal{B}_2^h$, where $\boldsymbol{\xi}$ is a subgradient and α is the corresponding linearization error instead of the triplets in (11) and (12).

In the beginning of Step 3, the bundles $\mathcal{B}_1^{h+1}$ and $\mathcal{B}_2^{h+1}$ can be freely chosen, and it is possible to reset either the bundle $\mathcal{B}_1^{h+1}$ or $\mathcal{B}_2^{h+1}$ or even both. However, both of these bundles must contain at least one element in Step 1. This is guaranteed, since at the end of Step 3, we add elements corresponding to the new iteration point $\boldsymbol{x}_{h+1}$ into both bundles.

Next we discuss about the main iteration of MDBDC in Algorithm 3. Since the current iteration point $\boldsymbol{x}_h$ does not change during the execution of Algorithm 3, we omit the index h, expect for $\boldsymbol{x}_h$, to simplify the algorithm. The execution of Algorithm 3 either yields a new iteration point or ensures Clarke stationarity of our current solution.

In practice, the sizes of the bundles are limited. The size of $\mathcal{B}_1$ has to be large enough to contain space for elements related to both $\mathcal{B}_A$ and $\mathcal{B}_B$ meaning that $|J_1| \geq k + m$. On the other hand, we can control the number of subproblems

Algorithm 2 Multiobjective double bundle method for DC functions (MDBDC)

Data: The stopping tolerance $\delta \in (0,1)$, the proximity measure $\varepsilon > 0$, the enlargement parameter $\theta > 0$, the quality measure $\eta \geq 0$, the decrease parameters $r, c_1, c_2, c_3 \in (0,1)$, the increase parameter $R > 1$, the descent parameters $m_1, m_2 \in (0,1)$ and $m_3 \in (m_2, 1)$, and the threshold $\tau_{\max} > 0$.

Step 0. *(Initialization)* Select $\boldsymbol{x}_0 \in X$ and execute Algorithm 4 for scaling. Compute $\boldsymbol{a}_i \in \partial A_i(\boldsymbol{x}_0, \boldsymbol{x}_0)$ for all $i \in I$, $\boldsymbol{b}_l \in \partial B_l(\boldsymbol{x}_0)$ for all $l \in L$, and $\boldsymbol{h}_2 \in \partial H_2(\boldsymbol{x}_0)$. Initialize $\mathcal{B}_1^0$ and $\mathcal{B}_2^0$ by setting $J_1^0 = J_2^0 = \{1\}$ and $\mathcal{B}_{A,i}^0 = \big\{(\boldsymbol{a}_i, 0)\big\}$ for all $i \in I$, $\mathcal{B}_{B,l}^0 = \big\{(\boldsymbol{b}_l, 0)\big\}$ for all $l \in L$, and $\mathcal{B}_2^0 = \big\{(\boldsymbol{h}_2, 0)\big\}$. Set $t = t_{\min} = t_{\max} = 0$. Initialize the counters $h = 0$ and $\tau = 0$.

Step 1. *(Main iteration)* Find a new iteration point $\boldsymbol{x}_{h+1}$ by executing Algorithm 3. If $\boldsymbol{x}_{h+1} = \boldsymbol{x}_h$, then Clarke stationarity is achieved, and STOP with $\boldsymbol{x}^* = \boldsymbol{x}_h$ as the final solution.

Step 2. *(Parameter update)* Initialize $\tilde{t} = t$.

(a) If $H(\boldsymbol{x}_{h+1}, \boldsymbol{x}_h) - H(\boldsymbol{x}_h, \boldsymbol{x}_h) \leq m_3\big(\hat{H}^h(\boldsymbol{x}_{h+1}) - H(\boldsymbol{x}_h, \boldsymbol{x}_h)\big)$ and $\tau > 0$, set

$$\tilde{t} = 0.5t\frac{\hat{H}^h(\boldsymbol{x}_{h+1}) - H(\boldsymbol{x}_h, \boldsymbol{x}_h)}{\hat{H}^h(\boldsymbol{x}_{h+1}) - H(\boldsymbol{x}_{h+1}, \boldsymbol{x}_h)}$$

and go to Step 2(c).

(b) If $\tau > 3$, then set $\tilde{t} = 2t$.

(c) Set $t^+ = \max\big\{\min\{\tilde{t}, 10t, t_{\max}\},\, t_{\min}\big\}$ and $\tau = \max\{1, \tau + 1\}$. If $t \neq t^+$, then update $t = t^+$ and $\tau = 1$.

Step 3. *(Bundle update)* Select the bundles $J_1^{h+1} \subseteq J_1^h$ and $J_2^{h+1} \subseteq J_2^h$. Update the linearization errors using (22) for all the elements in $\mathcal{B}_1^{h+1}$ and $\mathcal{B}_2^{h+1}$. Compute $\boldsymbol{a}_i \in \partial A_i(\boldsymbol{x}_{h+1}, \boldsymbol{x}_{h+1})$ for all $i \in I$, $\boldsymbol{b}_l \in \partial B_l(\boldsymbol{x}_{h+1})$ for all $l \in L$, and $\boldsymbol{h}_2 \in \partial H_2(\boldsymbol{x}_{h+1})$. Insert $(\boldsymbol{a}_i, 0)$ and $(\boldsymbol{b}_l, 0)$ into $\mathcal{B}_1^{h+1}$ for all $i \in I$, $l \in L$, and $(\boldsymbol{h}_2, 0)$ into $\mathcal{B}_2^{h+1}$. Select an index j corresponding to these insertions and add j into J_1^{h+1} and J_2^{h+1}. Set $h = h + 1$ and go to Step 1.

Algorithm 3 Main iteration for MDBDC

Data: The stopping tolerance $\delta \in (0,1)$, the enlargement parameter $\theta > 0$, the quality measure $\eta \geq 0$, the decrease parameters $r, c_1, c_2, c_3 \in (0,1)$, the increase parameter $R > 1$, the descent parameter $m_2 \in (0,1)$, the threshold $\tau_{\max} > 0$, and the subgradients $\boldsymbol{h}_1(\boldsymbol{x}_h) \in \partial H_1(\boldsymbol{x}_h, \boldsymbol{x}_h)$ and $\boldsymbol{h}_2(\boldsymbol{x}_h) \in \partial H_2(\boldsymbol{x}_h)$.

Step 0. *(Initialization)* Set $\boldsymbol{d}_t = \boldsymbol{0}$. Calculate $j^* = \operatorname{argmax}_{j \in J_2}\{\|\boldsymbol{h}_{2,j}\|\}$ and set $\boldsymbol{h}_{2,\max} = \boldsymbol{h}_{2,j^*}$,

$$t_{\min} = r \cdot \frac{\theta}{2(\|\boldsymbol{h}_1(\boldsymbol{x}_h)\| + \|\boldsymbol{h}_{2,\max}\|)}, \tag{23}$$

and $t_{\max} = Rt_{\min}$. If $t \notin [t_{\min}, t_{\max}]$, then select $t \in [t_{\min}, t_{\max}]$.

Step 1. *(Criticality)* If $\|\boldsymbol{h}_1(\boldsymbol{x}_h) - \boldsymbol{h}_2(\boldsymbol{x}_h)\| < \delta$, then go to Step 3.

Step 2. *(Search direction)* Calculate the search direction $\boldsymbol{d}_t$ as a solution of the problem (16).

Step 3. *(Clarke stationarity)* If $\|\boldsymbol{d}_t\| < \delta$ or $\hat{H}(\boldsymbol{x}_h + \boldsymbol{d}_t) - H(\boldsymbol{x}_h, \boldsymbol{x}_h) > -\eta$, then execute Algorithm 1 for the point $\boldsymbol{x}_h$. Set $\boldsymbol{x}_{h+1} = \boldsymbol{x}^+$ and $\tau = 0$, and EXIT.

Step 4. *(Descent test)* Set $\boldsymbol{y} = \boldsymbol{x}_h + \boldsymbol{d}_t$. If

$$H(\boldsymbol{y}, \boldsymbol{x}_h) - H(\boldsymbol{x}_h, \boldsymbol{x}_h) \leq m_2\big(\hat{H}(\boldsymbol{y}) - H(\boldsymbol{x}_h, \boldsymbol{x}_h)\big), \tag{24}$$

then set $\boldsymbol{x}_{h+1} = \boldsymbol{y}$ and EXIT.

Step 5. *(Bundle update)* Compute $\boldsymbol{a}_i \in \partial A_i(\boldsymbol{y}, \boldsymbol{x}_h)$ and $\alpha_i^A(\boldsymbol{x}_h, \boldsymbol{y})$ for all $i \in I$, $\boldsymbol{b}_l \in \partial B_l(\boldsymbol{y})$ and $\alpha_l^B(\boldsymbol{x}_h, \boldsymbol{y})$ for all $l \in L$, and $\boldsymbol{h}_2 \in \partial H_2(\boldsymbol{y})$ and $\alpha_2^H(\boldsymbol{x}_h, \boldsymbol{y})$.

(a) If $\boldsymbol{y} \notin \mathcal{F}_0$ and $\|\boldsymbol{d}_t\| > \theta$, then set $t = t - c_1(t - t_{\min})$ and $\tau = 0$. Go to Step 2.

(b) If $\tau \geq -\tau_{\max}$, then $t = t - c_2(t - t_{\min})$. Otherwise, $t = t - c_3(t - t_{\min})$. Set $\tau = \min\{-1, \tau - 1\}$. Insert $(\boldsymbol{a}_i, \alpha_i^A(\boldsymbol{x}_h, \boldsymbol{y}))$ and $(\boldsymbol{b}_l, \alpha_l^B(\boldsymbol{x}_h, \boldsymbol{y}))$ into $\mathcal{B}_1$ for all $i \in I$, $l \in L$, and $(\boldsymbol{h}_2, \alpha_2^H(\boldsymbol{x}_h, \boldsymbol{y}))$ into $\mathcal{B}_2$. Select a suitable index j corresponding to these insertions and add j into J_1 and J_2.

Step 6. *(Parameter update)* If $\|\boldsymbol{h}_2\| > \|\boldsymbol{h}_{2,\max}\|$, then set $\boldsymbol{h}_{2,\max} = \boldsymbol{h}_2$ and update $t_{\min}$ using (23). Go to Step 2.

solved in Step 2 being the most time-consuming part of Algorithm 3. The only restriction is that $(\boldsymbol{h}_2(\boldsymbol{x}_h), 0)$ must be included into $\mathcal{B}_2$, and thus $|J_2| \geq 1$.

Due to the DC decomposition of the improvement function, one objective may dominate the others and hide their effect if the magnitudes of objective function values differ a lot. To avoid this, MDBDC contains a scaling procedure presented in Algorithm 4. With this procedure, we obtain modified objective functions maintaining the same optima as the original objectives.

Algorithm 4 Scaling procedure

Step 1. Calculate $i^* = \arg\min \big\{ |f_i(\boldsymbol{x}_0)| \;\big|\; i \in I \big\}$.

Step 2. For each $i \in I$, search the value κ_i such that $10^{\kappa_i - 1} \leq |f_i(\boldsymbol{x}_0)| \leq 10^{\kappa_i}$. If $\kappa_i < 0$, then $\kappa_i = 0$.

Step 3. For each $i \in I$, set $\nu_i = \kappa_{i^*} - \kappa_i$. If $\nu_i \leq -2$, then $\nu_i = \nu_i + 1$. Set $\omega_i = 10^{\nu_i}$ and $f_i = \omega_i f_i$ being the scaled objective function.

3.4 Convergence

We devote this section to prove the convergence of MDBDC. This convergence analysis is divided into four lemmas and two theorems. Lemmas 3.7 and 3.8 are auxiliary results and Lemmas 3.6 and 3.9 can be summarized by saying that there does not exists any infinite cycle in Algorithm 3. Finally, in Theorem 3.10 we state that Algorithm 2 stops after a finite number of iterations and Theorem 3.11 considers the weak Pareto stationarity of the solution. Throughout the convergence analysis, we assume that $\mathbf{A1}$ and $\mathbf{A2}$ are valid. Additionally, for $\theta > 0$ we define the set $\mathcal{F}_\theta = \{\boldsymbol{x} \in \mathbb{R}^n \,|\, d(\boldsymbol{x}, \mathcal{F}_0) \leq \theta\}$, where $d(\boldsymbol{x}, \mathcal{F}_0) = \inf\{\|\boldsymbol{x} - \boldsymbol{z}\| \,|\, \boldsymbol{z} \in \mathcal{F}_0\}$.

We begin recalling Theorem 4.9 in [16] asserting a finite upper bound for the number of iterations of Algorithm 1.

Lemma 3.6. *[16] Let the assumption* $\mathbf{A1}$ *be valid. Algorithm 1 terminates after at most*

$$\left\lceil \frac{4}{(1-m_1)^2} \left(\frac{K}{\delta}\right)^4 \right\rceil$$

iterations where $\lceil \cdot \rceil$ *is a ceiling of the number and* $K > 0$ *is the Lipschitz constant of* $H(\,\cdot\,, \boldsymbol{y})$ *at the point* $\boldsymbol{x} \in \mathbb{R}^n$*, when* $\boldsymbol{y} = \boldsymbol{x}$*.*

The following auxiliary result is proved in the same way as Lemma 3 in [15].

Lemma 3.7. *If the condition* (24) *in Step 4 of Algorithm 3 is not satisfied, then*

$$\boldsymbol{\xi}_1^T \boldsymbol{d}_t - \alpha_1 > m_2\big(\hat{H}_1(\boldsymbol{y}) - H_1(\boldsymbol{x}_h, \boldsymbol{x}_h)\big) + (1 - m_2)\big(\hat{H}_2(\boldsymbol{y}) - H_2(\boldsymbol{x}_h)\big),$$

where $\boldsymbol{\xi}_1 \in \partial H_1(\boldsymbol{y}, \boldsymbol{x}_h)$ *is a subgradient calculated at the new auxiliary point* $\boldsymbol{y} = \boldsymbol{x}_h + \boldsymbol{d}_t$ *and* $\alpha_1 = H_1(\boldsymbol{x}_h, \boldsymbol{x}_h) - H_1(\boldsymbol{y}, \boldsymbol{x}_h) + \boldsymbol{\xi}_1^T \boldsymbol{d}_t$ *is the corresponding linearization error.*

Proof. If the condition (24) is not satisfied in Step 4, then

$$H(\boldsymbol{y}, \boldsymbol{x}_h) - H(\boldsymbol{x}_h, \boldsymbol{x}_h) > m_2(\hat{H}(\boldsymbol{y}) - H(\boldsymbol{x}_h, \boldsymbol{x}_h)).$$

By utilizing Lemma 3.2 (i) and the DC decomposition of $H(\,\cdot\,, \boldsymbol{x}_h)$, we get

$$\begin{aligned} H_1(\boldsymbol{y}, \boldsymbol{x}_h) - H_1(\boldsymbol{x}_h, \boldsymbol{x}_h) >& m_2(\hat{H}(\boldsymbol{y}) - H(\boldsymbol{x}_h, \boldsymbol{x}_h)) + H_2(\boldsymbol{y}) - H_2(\boldsymbol{x}_h) \\ \geq& m_2(\hat{H}(\boldsymbol{y}) - H(\boldsymbol{x}_h, \boldsymbol{x}_h)) + \hat{H}_2(\boldsymbol{y}) - H_2(\boldsymbol{x}_h) \end{aligned}$$

We obtain the result by noticing that

$$H_1(\boldsymbol{y}, \boldsymbol{x}_h) - H_1(\boldsymbol{x}_h, \boldsymbol{x}_h) = \boldsymbol{\xi}_1^T \boldsymbol{d}_t - \alpha_1,$$

when $\boldsymbol{\xi}_1 \in \partial H_1(\boldsymbol{y}, \boldsymbol{x}_h)$ and $\alpha_1 = H_1(\boldsymbol{x}_h, \boldsymbol{x}_h) - H_1(\boldsymbol{y}, \boldsymbol{x}_h) + \boldsymbol{\xi}_1^T \boldsymbol{d}_t$. □

Before stating the finite convergence of Algorithm 3, we collect some crucial observations.

Lemma 3.8. *Let the assumption* **A2** *be valid. During each execution of Algorithm 3*

(i) the points $\boldsymbol{x}_h$ *and* $\boldsymbol{y}_j$ *for all* $j \in J_1 \cup J_2$ *belong to the set* $\mathcal{F}_\theta$.

(ii) there exists $C > 0$ *such that* $\|\boldsymbol{x}_h - \boldsymbol{y}_j\| < C$ *for all* $\boldsymbol{y}_j \in \mathcal{B}_1 \cup \mathcal{B}_2$.

(iii) the subgradients $\boldsymbol{a}_{i,j}$ *and* $\boldsymbol{b}_{l,j}$ *and the linearization errors* $\alpha_{i,j}^A$ *and* $\alpha_{l,j}^B$ *for all* $i \in I$, $l \in L$, *and* $j \in J_1$ *are bounded.*

(iv) the subgradients $\boldsymbol{h}_{2,j}$ *and the linearization errors* $\alpha_{2,j}^H$ *for all* $j \in J_2$ *are bounded.*

(v) $t_{\min}$ *is bounded from below with a positive threshold and* $t_{\max}$ *is bounded from above.*

Proof. (i) The points $\boldsymbol{y}_j$ on $\mathcal{B}_1$ and $\mathcal{B}_2$ are ensured to belong to the set $\mathcal{F}_\theta$ by Step 5 of Algorithm 3. In addition, the point $\boldsymbol{x}_h$ is on the set $\mathcal{F}_\theta$, since each new iteration point decreases the value of the objectives.
(ii) From the assumption **A2**, we obtain that the set $\mathcal{F}_\theta$ is compact, and together with (i), there exists a constant $C > 0$ such that $\|\boldsymbol{x}_h - \boldsymbol{y}_j\| < C$ for all $\boldsymbol{y}_j \in \mathcal{B}_1 \cup \mathcal{B}_2$.
(iii) Every $A_i(\,\cdot\,, \boldsymbol{x}_h)$ and B_l for all $i \in I$ and $l \in L$ are convex implying their local Lipschitz continuity. Thus, there exist a Lipschitz constant for each of these functions on the compact set $\mathcal{F}_\theta$. Let $K_1 > 0$ be the constant overestimating all these constants. The property (i) yields $\|\boldsymbol{a}_{i,j}\| \leq K_1$ and $\|\boldsymbol{b}_{l,j}\| \leq K_1$ for all $i \in I$,

$l \in L$ and $j \in J_1$. Combining (ii) with the above observations, we deduce for all $i \in I$ and $j \in J_1$ that

$$\begin{aligned}|\alpha^A_{i,j}| &= |A_i(\boldsymbol{x}_h, \boldsymbol{x}_h) - A_i(\boldsymbol{y}_j, \boldsymbol{x}_h) - \boldsymbol{a}^T_{i,j}(\boldsymbol{x}_h - \boldsymbol{y}_j)| \\ &\le |A_i(\boldsymbol{x}_h, \boldsymbol{x}_h) - A_i(\boldsymbol{y}_j, \boldsymbol{x}_h)| + \|\boldsymbol{a}_{i,j}\|\|\boldsymbol{x}_h - \boldsymbol{y}_j\| \\ &\le K_1\|\boldsymbol{x}_h - \boldsymbol{y}_j\| + K_1 C \le 2K_1 C.\end{aligned}$$

Similarly, we can show that $|\alpha^B_{l,j}| \le 2K_1C$ for every $l \in L$ and $j \in J_1$.
(iv) The function H_2 is convex and thus, locally Lipschitz continuous. Therefore, we have a Lipschitz constant $K_2 > 0$ for H_2 on the compact set $\mathcal{F}_\theta$. As in (iii), we can show that $\|\boldsymbol{h}_{2,j}\| \le K_2$ and $|\alpha^H_{2,j}| \le 2K_2C$ for every $j \in J_2$.
(v) From (iii) and (iv) we can derive that

$$t_{\min} \ge \bar{t}_{\min} = \frac{r\theta}{2(K_1 + K_2)} > 0$$

yielding the positive lower bound for $t_{\min}$. If the condition in Step 1 of Algorithm 3 is not satisfied, then

$$\delta \le \|\boldsymbol{h}_1(\boldsymbol{x}_h) - \boldsymbol{h}_2(\boldsymbol{x}_h)\| \le \|\boldsymbol{h}_1(\boldsymbol{x}_h)\| + \|\boldsymbol{h}_2(\boldsymbol{x}_h)\| \le \|\boldsymbol{h}_1(\boldsymbol{x}_h)\| + \|\boldsymbol{h}_{2,\max}\|.$$

The upper bound for $t_{\max}$ is obtained, since

$$t_{\max} \le \bar{t}_{\max} = \frac{Rr\theta}{2\delta} < \infty. \tag{25}$$

□

Next we are in a position to show the same kind of a result as Theorem 5 in [15] guaranteeing that we do not have an infinite loop in Algorithm 3.

Lemma 3.9. *Let the assumption* **A2** *be valid. For any* $\delta \in (0, 1)$*, Algorithm 3 cannot pass infinitely through the sequence of Steps from 2 to 6.*

Proof. Suppose that the assertion is false and Steps from 2 to 6 are executed infinitely. We index by $i \in \mathcal{I}$ the quantities related to the i-th passage. Since Algorithm 1 cannot ever be entered, the condition $\|\boldsymbol{d}^i_t\| \ge \delta$ is satisfied for each $i \in \mathcal{I}$.

Assume that Step 5(a) is passed infinitely. In Step 5, the proximity parameter t decreases at each pass and converges to $t_{\min}$, since $t_{\min}$ is monotonically decreasing and by Lemma 3.8 (v), it is bounded from below. In addition, $t_{\min}$ is always smaller than the threshold

$$\rho = \frac{\theta}{2\big(\|\boldsymbol{h}_1(\boldsymbol{x}_h)\| + \|\boldsymbol{h}^i_{2,\max}\|\big)},$$

and therefore, after a finite number of passes, t will be below ρ. This yields a contradiction, since $\|\boldsymbol{d}_t^i\| \leq \theta$ by Lemma 3.3 and Step 5(a) can no longer be executed.

Due to Lemma 3.3, the parameter selection rule $t \in [t_{\min}, t_{\max}]$, and Lemma 3.8 (iii), (iv) and (v), the sequence $\{\boldsymbol{d}_t^i\}_{i\in\mathcal{I}}$ is bounded. Thus, there exists a convergent subsequence $\{\boldsymbol{d}_t^i\}_{i\in\mathcal{I}'\subseteq\mathcal{I}}$. Additionally, by combining Lemma 3.8 (iii) and (iv), the sequences $\{\hat{H}_1(\boldsymbol{y}^i)\}_{i\in\mathcal{I}'\subseteq\mathcal{I}}$ and $\{\hat{H}_2(\boldsymbol{y}^i)\}_{i\in\mathcal{I}'\subseteq\mathcal{I}}$ are bounded. Hence, these sequences have the convergent subsequences for $i \in \mathcal{I}'' \subseteq \mathcal{I}'$ and their limits are denoted by $\hat{H}_1^*$ and $\hat{H}_2^*$, respectively. From Lemma 3.2 (ii) and $\|\boldsymbol{d}_t^i\| \geq \delta$, we obtain for all $i \in \mathcal{I}$

$$\hat{H}_1(\boldsymbol{y}^i) - \hat{H}_2(\boldsymbol{y}^i) - H(\boldsymbol{x}_h, \boldsymbol{x}_h) \leq -\frac{1}{2t_i}\|\boldsymbol{d}_t^i\|^2 \leq -\frac{\delta^2}{2t_i} < 0. \tag{26}$$

Let $t^* = \lim_{i\to\infty} t_i$. Now $t^* > 0$ exists, since the sequence t_i is nonincreasing and bounded from below with a positive threshold by Lemma 3.8 (v) and the selection $t \in [t_{\min}, t_{\max}]$. Finally, passing to the limit in (26) yields

$$\hat{H}_1^* - \hat{H}_2^* - H(\boldsymbol{x}_h, \boldsymbol{x}_h) \leq -\frac{\delta^2}{2t^*} < 0. \tag{27}$$

To obtain a contradiction, we consider two successive indices v and w in $\mathcal{I}''$ and let $\alpha_{1,v} = H_1(\boldsymbol{x}_h, \boldsymbol{x}_h) - H_1(\boldsymbol{y}^v, \boldsymbol{x}_h) + \boldsymbol{\xi}_{i,v}^T \boldsymbol{d}_t^v$, where $\boldsymbol{\xi}_{i,v} \in \partial H_1(\boldsymbol{y}^v, \boldsymbol{x}_h)$. Now Lemma 3.7 gives

$$\boldsymbol{\xi}_{1,v}^T \boldsymbol{d}_t^v - \alpha_{1,v} > m_2\big(\hat{H}_1(\boldsymbol{y}^v) - H_1(\boldsymbol{x}_h, \boldsymbol{x}_h)\big) + (1 - m_2)\big(\hat{H}_2(\boldsymbol{y}^v) - H_2(\boldsymbol{x}_h)\big)$$

and utilizing the definition of $\hat{H}_1$, we get

$$\hat{H}_1\big(\boldsymbol{y}^w\big) - H_1(\boldsymbol{x}_h, \boldsymbol{x}_h) \geq \boldsymbol{\xi}_{1,v}^T \boldsymbol{d}_t^w - \alpha_{i,v}.$$

By combining these two inequalities we conclude

$$\boldsymbol{\xi}_{1,v}^T\big(\boldsymbol{d}_t^v - \boldsymbol{d}_t^w\big) > m_2\hat{H}_1\big(\boldsymbol{y}^v\big) - \hat{H}_1\big(\boldsymbol{y}^w\big) + (1 - m_2)\big(\hat{H}_2\big(\boldsymbol{y}^v\big) + H(\boldsymbol{x}_h, \boldsymbol{x}_h)\big).$$

A passage to the limit yields

$$(m_2 - 1)\big(\hat{H}_1^* - \hat{H}_2^* - H(\boldsymbol{x}_h, \boldsymbol{x}_h)\big) < 0,$$

but since $m_2 \in (0, 1)$, the property (27) cannot hold. □

Summarizing, we have now considered all the possibilities where the infinite cycle may happen in Algorithm 3. We have thus led to the following theorem stating the finite convergence of MDBDC.

Theorem 3.10. *Let the assumptions* **A1** *and* **A2** *be valid. For any* $\delta \in (0, 1)$ *and* $\varepsilon > 0$*, the execution of Algorithm 2 stops after a finite number of iterations at the point* $\boldsymbol{x}^*$ *satisfying the approximate Clarke stationary condition* $\|\boldsymbol{\xi}^*\| \leq \delta$*, where* $\boldsymbol{\xi}^* \in \partial_\varepsilon^G H(\boldsymbol{x}^*, \boldsymbol{x}^*)$*.*

Proof. The execution of Algorithm 2 can stop only if the Clarke stationary point $\boldsymbol{x}^*$ is found in Step 1 meaning that the approximate Clarke stationary condition is satisfied in Step 1 of Algorithm 1. Assume that Algorithm 2 is executed infinitely. By Lemmas 3.6 and 3.9, the new iteration point $\boldsymbol{x}_{h+1}$ is obtained after a finite number of iterations in Step 3 or 4 of Algorithm 3. Therefore, we obtain a sequence of solutions $\{\boldsymbol{x}_h\}$, where

$$\boldsymbol{x}_h = \boldsymbol{x}_{h-1} + \sigma \boldsymbol{d}_h \quad \text{such that} \quad \sigma \geq \min\{1, \varepsilon\} > 0. \tag{28}$$

Moreover, the sequence $\{\boldsymbol{x}_h\}$ belongs to the level set $\mathcal{F}_0$ being compact by $\mathbf{A2}$, and thus the sequence $\{\boldsymbol{x}_h\}$ has an accumulation point $\bar{\boldsymbol{x}}$. Since $\{\boldsymbol{x}_h\} \to \bar{\boldsymbol{x}}$ and $\boldsymbol{x}_h$ has the formulation (28), it follows that the sequence $\{\boldsymbol{d}_h\} \to \mathbf{0}$. Consequently, $\|\boldsymbol{d}_h\| \to 0$ and especially for any $\delta \in (0, 1)$ there exists an iteration index h' such that $\|\boldsymbol{d}_h\| < \delta$ for all $h \geq h'$.

Due to this, during the iteration h' we cannot obtain a new point in Step 4 of Algorithm 3, and thus we obtain a new point in Step 3 of Algorithm 3. However, the search direction selection in Step 2 of Algorithm 1 yields $\|\boldsymbol{d}_{h'}\| = 1 > \delta$ being a contradiction, since $\|\boldsymbol{d}_{h'}\| < \delta$. □

Finally, we guarantee the weak Pareto stationarity of the solution with a similar result than Theorem 7 in [25].

Theorem 3.11. *Let f_i and g_l be DC functions for all $i \in I$ and $l \in L$. Suppose that $G^{\leq}(\boldsymbol{x}) \subseteq K_X(\boldsymbol{x})$ and the assumptions* $\mathbf{A1}$ *and* $\mathbf{A2}$ *are valid. Then, MDBDC stops after a finite number of iterations with the solution $\boldsymbol{x}^*$ being a weakly Pareto stationary point for the problem* (1).

Proof. Consider a single objective unconstrained minimization problem with the improvement function $H(\,\cdot\,, \boldsymbol{x})$ as its objective. According to Theorem 3.10, after a finite number of iterations, MDBDC finds a solution $\boldsymbol{x}^* \in \mathbb{R}^n$ such that it is a Clarke stationary point for the improvement function $H(\,\cdot\,, \boldsymbol{x}^*)$ yielding $\mathbf{0} \in \partial H(\boldsymbol{x}^*, \boldsymbol{x}^*)$. Thus, by Theorem 3.1 the solution $\boldsymbol{x}^*$ is weak Pareto stationary for the problem (1). □

4 Numerical experiments

In this section, we study the behaviour of MDBDC. We have collected some academic single-objective DC problems and combined those to obtain multiobjective DC problems. The problems obtained are solved with MDBDC and MPB [25, 28]. The aim of these numerical tests is, on the one hand, to verify the usability of MDBDC in practice, and on the other hand, to justify the use of the DC method instead of the general nonconvex method.

The implementation of MPB is done with Fortran 77 and it is described in [23]. MDBDC is implemented with Fortran 95 and both implementations utilize the quadratic solver by Lukšan described in [22]. Additionally, MPB applies an aggregation strategy [17, 18]. The tests are performed under Linux Ubuntu system and `f95` is used as a compiler. We remark that in Step 6 of Algorithm 1 we update $\tilde{\boldsymbol{x}}$ if the step-length $\beta^* < \varepsilon$. However, in practice, the implementation of MDBDC stops at this point and the final solution is $\tilde{\boldsymbol{x}}$.

The input parameters for MDBDC are chosen as follows: the stopping tolerance $\delta = 10^{-5}$, the proximity measure $\varepsilon = 10^{-4}$, the enlargement parameter $\theta = 5 \cdot 10^{-5}$, the quality measure and the decrease parameters

$$\eta = \begin{cases} 0, & \text{if } n \leq 100 \\ 10^{-5}, & \text{if } 100 < n \leq 300 \\ 10^{-4}, & \text{if } n > 300 \end{cases}, \quad r = \begin{cases} 0.75, & \text{if } n < 10 \\ \frac{n}{n+5}, & \text{if } 10 \leq n \leq 300 \\ 0.99, & \text{if } n > 300 \end{cases},$$

$$c = \begin{cases} 0.4, & \text{if } n < 25 \\ 0.25, & \text{if } 25 \leq n < 100 \\ 0.1, & \text{if } 100 \leq n < 200 \\ 0.01, & \text{if } 200 \leq n < 300 \\ 0.001, & \text{if } n \geq 300 \end{cases}$$

$c_1 = 0.5$, $c_2 = \min\{0.5, c \cdot (k-1)\}$ and $c_3 = 0.1$, the increase parameter $R = 10^{10}$, the descent parameters $m_1 = m_2 = 0.01$ and $m_3 = 0.1$, and the threshold $\tau_{\max} = 50$. The maximum size for the bundle $\mathcal{B}_1$ is $\min\{(n+5) \cdot (k+m), 1000\}$ and for the bundle $\mathcal{B}_2$ it is 3. The size of U_j in Algorithm 1 is $2(n+5)$. Note that MDBDC is quite sensitive for the parameter selection, and by specifying the parameters for the problem, the execution times of MDBDC may improve a lot. The parameters selected for MPB are default values [23].

The objective functions of the test problems are described in Table 1. The constraint functions of the form $g = r - s \leq 0$ are

$$\begin{aligned} \mathbf{C1}: \quad & r(\boldsymbol{x}) = \max\{(x_1+1.5)^2 + (x_1-1)^2 + x_2^2 + (x_2-1)^2 - 5,\ 0\}, \\ & s(\boldsymbol{x}) = (x_1-1)^2 + (x_2-1)^2 - 1 \\ \mathbf{C2}: \quad & r(\boldsymbol{x}) = 0, \\ & s(\boldsymbol{x}) = \max\{x_1^2 + x_2^2 + x_3^2 + x_4^2 - 10,\ x_1 + x_2 + x_3 + x_4 - 5.5\} \\ \mathbf{C3}: \quad & r(\boldsymbol{x}) = 0.5n, \\ & s(\boldsymbol{x}) = \sum_{i=1}^{n} \left(\boldsymbol{x}_i + (-1)^{i+1} \cdot 0.5\right)^2 \end{aligned}$$

Note that **C1** is a DC constraint and both **C2** and **C3** are concave. Finally, the unconstrained and the constrained multiobjective test problems are described in Tables 2 and 3, respectively.

The results of the tests performed are reported in Table 4. The problems 1–15 are unconstrained and the problems 16–21 are constrained. In Table 4, the first

Table 1: Objective functions, $\mathbf{O1} - \mathbf{O7}$ from [15] and $\mathbf{O8} - \mathbf{O12}$ from [16]

Objective	Function	Objective	Function	Objective	Function
$\mathbf{O1}$	Problem 2.	$\mathbf{O5}$	Problem 7.	$\mathbf{O9}$	Problem 13.
$\mathbf{O2}$	Problem 3.	$\mathbf{O6}$	Problem 9.	$\mathbf{O10}$	Problem 14.
$\mathbf{O3}$	Problem 4.	$\mathbf{O7}$	Problem 10.	$\mathbf{O11}$	Problem 15.
$\mathbf{O4}$	Problem 6.	$\mathbf{O8}$	Problem 12.	$\mathbf{O12}$	Problem 16.

Table 2: Unconstrained testproblems, $i = 1, \ldots, n$

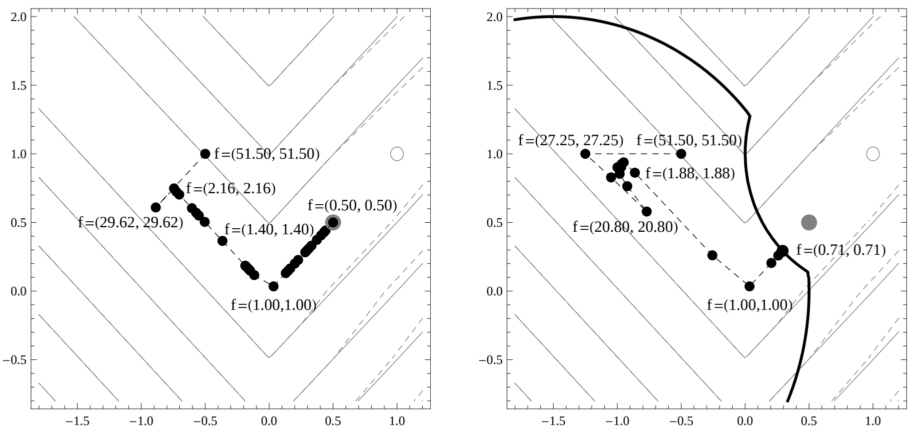
P	f_1	f_2	$\boldsymbol{x}_0$	P	f_1	f_2	$\boldsymbol{x}_0$	P	f_1	f_2	f_3	$\boldsymbol{x}_0$
1.	$\mathbf{O1}$	$\mathbf{O4}$	$(-1.2, 1)$	6.	$\mathbf{O7}$	$\mathbf{O9}$	$x_0^i = 0.1i$	11.	$\mathbf{O1}$	$\mathbf{O4}$	$\mathbf{O5}$	$(-1.2, 1)$
2.	$\mathbf{O1}$	$\mathbf{O5}$	$(-0.5, 1)$	7.	$\mathbf{O8}$	$\mathbf{O9}$	$x_0^i = 2i$	12.	$\mathbf{O2}$	$\mathbf{O3}$	$\mathbf{O6}$	$(1, 3, 3, 1)$
3.	$\mathbf{O1}$	$\mathbf{O5}$	$(-1.2, 1)$	8.	$\mathbf{O3}$	$\mathbf{O7}$	$x_0^i = 0.1i$	13.	$\mathbf{O3}$	$\mathbf{O7}$	$\mathbf{O8}$	$x_0^i = 0.1i$
4.	$\mathbf{O4}$	$\mathbf{O5}$	$(-2, 1)$	9.	$\mathbf{O7}$	$\mathbf{O8}$	$x_0^i = 2i$	14.	$\mathbf{O3}$	$\mathbf{O7}$	$\mathbf{O12}$	$x_0^i = 0.1i$
5.	$\mathbf{O2}$	$\mathbf{O6}$	$(4, 2, 4, 2)$	10.	$\mathbf{O10}$	$\mathbf{O11}$	$x_0^i = (-1)^{i+1}$	15.	$\mathbf{O7}$	$\mathbf{O10}$	$\mathbf{O11}$	$x_0^i = 0.1i$

Table 3: Testproblems with constraints, $i = 1, \ldots, n$

Problem	f_1	f_2	g_1	$\boldsymbol{x}_0$	Problem	f_1	f_2	f_3	g_1	$\boldsymbol{x}_0$
16.	$\mathbf{O1}$	$\mathbf{O5}$	$\mathbf{C1}$	$(-0.5, 1)$	19.	$\mathbf{O1}$	$\mathbf{O4}$	$\mathbf{O5}$	$\mathbf{C1}$	$(-1.2, 1)$
17.	$\mathbf{O2}$	$\mathbf{O6}$	$\mathbf{C2}$	$(4, 2, 4, 2)$	20.	$\mathbf{O2}$	$\mathbf{O3}$	$\mathbf{O6}$	$\mathbf{C2}$	$(1, 3, 3, 1)$
18.	$\mathbf{O7}$	$\mathbf{O8}$	$\mathbf{C3}$	$x_0^i = 2i$	21.	$\mathbf{O3}$	$\mathbf{O7}$	$\mathbf{O12}$	$\mathbf{C3}$	$x_0^i = 0.1i$

column describes the problem solved and n is the dimension of the problem. In order to compare the methods, we have given the number of function calls n_f, the number of subgradient evaluations $n_{\boldsymbol{\xi}}$ and the CPU time. Since for MPB $n_f = n_{\boldsymbol{\xi}}$, only $n_{\boldsymbol{\xi}}$ is reported in addition to the CPU time. In practice, n_f and $n_{\boldsymbol{\xi}}$ tell the number of function values and subgradients evaluated for each objective and constraint. Lastly, the column $f(\boldsymbol{x}^*) = (f_1(\boldsymbol{x}^*), \ldots, f_k(\boldsymbol{x}^*))$ describes the solution obtained.

Two example executions of MDBDC are illustrated in Figure 1. In these figures, dashed gray contours correspond to $\mathbf{O1}$ and the gray contours correspond to $\mathbf{O5}$. The optimum of the objective $\mathbf{O1}$ is at the point $\boldsymbol{x}^* = (0.50, 0.50)$ marked with a gray disk, and the optimum of the objective $\mathbf{O5}$ is at the point $\boldsymbol{x}^* = (1.00, 1.00)$ marked with a gray circle. The black curve in Figure 1b presents the constraint $\mathbf{C1}$. In Figure 1a, we obtain a solution $\boldsymbol{x}^* = (0.50, 0.50)$ such that $f(\boldsymbol{x}^*) = (0.50, 0.50)$ being an individual optimum of the objective $\mathbf{O1}$ as well. In Figure 1b, we have added the constraint $\mathbf{C1}$ such that neither of the optima of the individual objectives is feasible. Now we get a solution $\boldsymbol{x}^* = (0.29, 0.29)$ and $f(\boldsymbol{x}^*) = (0.71, 0.71)$. This solution lies on the same line than where the



(a) Test problem 2 (b) Test problem 16

Figure 1: The performance of MDBDC in the decision space

individual optima are and the constraint is active.

In Table 4, test problems 2 and 3 are the same problem with different starting points. In test problem 2, we notice that MDBDC finds a solution having better values for both objectives than the solution obtained with MPB. However, if we change a starting point a little, like in test problem 3, both methods find equally good solutions. In general, we say that one solution is better than the other if it has better values for all the objectives. Even if both MDBDC and MPB find a weakly Pareto stationary solution, one might find a better solution. Reason for this is a nonconvex feasible set in objective space, since both local and global optima satisfy the Pareto stationarity condition (7). In the test problems performed, MDBDC obtains a better solution in 16 cases, the solutions are equally good in 36 cases, and both methods fail in test problem 10 with $n = 500$. The better solutions obtained are bolded in Table 4.

In the computational point of view, MDBDC is a good alternative for MPB when objectives and constraints are DC functions, even though it may sometimes require more computational efforts. For instance, in test problem 8, MDBDC uses more function and subgradient evaluations in the cases where $n = 10$ and $n = 100$. However, compared with MPB, the solutions obtained with MDBDC are better in both of those cases. Another example about this kind of behaviour is seen in Figure 1. In both test problems 2 and 16, we obtain a solution $\boldsymbol{x}^* = (0.00, 0.00)$ and $f(\boldsymbol{x}^*) = (1.00, 1.00)$ with MPB. As we see, in both of those cases MDBDC visits also this point but can still continue forward. Generally speaking about the computational efforts of MDBDC in the test problems, we notice that MDBDC uses less function and subgradient evaluations in 26 cases, the number of evaluations are on the same magnitude in 6 cases, and MPB uses less evaluations in 20 cases. However, in these 20 test problems, MDBDC finds a better solution in half of the cases.

Table 4: Numerical results for unconstrained test problems

P	n	MDBDC n_f	MDBDC n_ξ	MDBDC CPU	MDBDC $f(\boldsymbol{x}^*)$	MPB n_ξ	MPB CPU	MPB $f(\boldsymbol{x}^*)$
1.	2	9	9	0.00	$(16.8643, -1.3156)^*$	6	0.00	$(21.6036, -1.3385)$
2.	2	122	94	0.01	$(\mathbf{0.5000}, \mathbf{0.5001})$	21	0.00	$(1.0000, 1.0000)$
3.	2	84	58	0.01	$(0.5000, 0.5000)$	42	0.00	$(0.4896, 0.5938)$
4.	2	9	9	0.00	$(-1.6851, 100.2572)^*$	6	0.00	$(-1.6948, 102.6963)$
5.	4	30	26	0.00	$(\mathbf{99.0578}, \mathbf{5.4019})^*$	5	0.00	$(267.7000, 9.2000)$
6.	10	241	194	0.04	$(-1.6099, 0.0026)^*$	62	0.00	$(-3.1776, 2.2843)$
7.	10	175	165	0.06	$(42.3771, 14.9875)^*$	130	0.01	$(70.3498, 8.6848)$
8.	10	124	107	0.03	$(\mathbf{0.2185}, \mathbf{-2.1928})$	18	0.00	$(0.4557, -0.5734)$
	50	158	156	0.40	$(0.6131, -16.3563)$	419	0.52	$(3.6075, -39.7421)$
	100	122	105	0.86	$(\mathbf{10.0001}, \mathbf{-89.5000})^*$	79	0.10	$(58.0000, -41.5000)$
	250	289	207	14.59	$(\mathbf{28.0033}, \mathbf{-221.5000})^*$	541	6.58	$(79.0000, -170.5000)^{**}$
	500	192	160	14.29	$(\mathbf{26.0017}, \mathbf{-473.5000})^*$	474	8.93	$(458.4319, -32.4898)^{**}$
9.	10	77	68	0.02	$(-5.4392, 39.2907)^*$	431	0.01	$(-7.9504, 50.3218)$
	50	96	93	0.14	$(\mathbf{-39.3532}, \mathbf{251.2447})^*$	971	0.43	$(-23.3744, 369.9386)^{**}$
	100	287	252	2.09	$(\mathbf{-80.8056}, \mathbf{322.3542})^*$	7	0.00	$(-0.5000, 995.0000)$
	250	233	221	3.86	$(\mathbf{-215.8172}, \mathbf{1454.0982})^*$	3796	142.28	$(-204.2314, 2145.0655)^{**}$
	500	203	167	4.39	$(\mathbf{-439.1831}, \mathbf{3404.4254})^*$	9	0.02	$(-2.4998, 4992.0308)$
10.	10	1554	1374	0.53	$(0.0001, 2.3712)^*$	332	0.01	$(0.0001, 22.2879)$
	50	249	202	0.50	$(2.5772, 0.2423)^*$	151	0.13	$(0.0006, 65.8711)$
	100	665	625	15.39	$(3.5020, 1.5277)^*$	374	1.69	$(0.0068, 173.4411)^{**}$
	250	753	710	73.44	$(0.6288, 390.4155)$	837	48.98	$(0.0020, 422.7423)$
	500				fail			fail
11.	2	9	9	0.00	$(16.8643, -1.3156, 16.8643)^*$	6	0.00	$(21.5747, -1.3383, 21.5747)$
12.	4	99	81	0.02	$(176.9992, 1.7910, 3.8972)^*$	24	0.00	$(119.0604, 1.1995, 5.3975)$
13.	10	204	173	0.07	$(\mathbf{0.2041}, \mathbf{-2.7203}, \mathbf{62.5866})^*$	61	0.00	$(0.5737, -2.0108, 82.5078)$
	50	190	186	0.76	$(17.6105, 70.1618, 212.9270)^*$	664	1.80	$(1.2173, -28.9312, 297.4673)$
	100	135	133	1.83	$(25.0123, -76.1151, 855.2446)^*$	337	1.22	$(41.5649, -51.8967, 746.3927)$
	250	34	30	0.25	$(75.6912, -195.3250, 1849.8487)^*$	3703	193.07	$(64.4847, -188.3828, 1291.7758)$
	500	56	51	1.03	$(159.6711, -388.4394, 3530.6517)^*$	5747	1586.86	$(100.7987, -383.5040, 3360.6881)$
14.	10	323	320	0.12	$(1.4588, -1.1710, 1.7804 \cdot 10^{-9})$	40	0.00	$(2.2110, 0.8130, 8.5602 \cdot 10^{-6})$
	50	17	15	0.01	$(14.9550, 6.4469, 2.8916 \cdot 10^{-6})$	38	0.01	$(12.4717, 1.4674, 3.7110 \cdot 10^{-6})$
	100	35	35	0.06	$(48.3652, -1.3052, 8.0459 \cdot 10^{-7})^*$	33	0.01	$(67.7079, -11.5448, 9.2342)$
	250	148	146	3.01	$(54.7364, -68.7425, 4.5003 \cdot 10^{-6})^*$	847	25.34	$(49.7191, -100.3897, 4.3898)$
	500	199	196	12.10	$(128.3886, -173.7699, 9.2844 \cdot 10^{-6})^*$	1780	411.99	$(97.4854, -189.2866, 4.6295)$
15.	10	2226	1981	0.76	$(-3.2627, 6.5823 \cdot 10^{-7}, 3.4899)^*$	381	0.02	$(-3.2981, 2.9205 \cdot 10^{-5}, 7.7348)$
	50	2168	2156	8.99	$(\mathbf{-9.1827}, \mathbf{0.0001}, \mathbf{168.5418})^*$	406	0.20	$(1.5767, 0.0015, 176.7342)$
	100	327	317	7.57	$(\mathbf{-53.3365}, \mathbf{0.0002}, \mathbf{103.4822})^*$	723	1.85	$(11.7025, 0.0083, 479.0823)$
	250	183	178	8.89	$(-127.6345, 0.0003, 2537.3443)^*$	6764	84.86	$(97.2280, 0.0115, 1764.0444)$
	500	1077	1027	208.49	$(-157.4916, 0.0070, 118499.9671)^*$	28663	2021.67	$(766.3614, 0.0351, 112639.2066)$
16.	2	44	38	0.01	$(\mathbf{0.7071}, \mathbf{0.7071})$	47	0.00	$(1.0000, 1.0000)$
17.	4	147	136	0.03	$(\mathbf{33.7271}, \mathbf{8.6481})^*$	10	0.00	$(305.8642, 9.2000)$
18.	10	24	24	0.01	$(-8.4586, 82.2429)^*$	568	0.02	$(-8.4606, 82.5665)$
	50	65	65	0.17	$(-47.3604, 336.9333)^*$	1629	0.80	$(-48.4003, 446.9219)$
	100	50	50	0.17	$(-97.8782, 824.9328)^*$	3065	6.43	$(-98.1723, 870.7725)$
	250	265	151	2.60	$(-241.5555, 1915.6674)^*$	10973	259.93	$(-247.1970, 2102.8978)$
	500	122	64	1.03	$(-491.2862, 4014.5727)^*$	14744	1043.34	$(-496.5222, 4519.2576)$
19.	2	12	12	0.00	$(18.6552, -1.2756, 18.6552)^*$	11	0.00	$(21.8166, -1.2924, 21.8166)$
20.	4	54	38	0.01	$(173.2068, 1.7467, 4.0270)^*$	16	0.00	$(161.6051, 1.5937, 5.5697)$
21.	10	22	20	0.00	$(2.9017, -0.5934, 9.1556 \cdot 10^{-7})$	68	0.00	$(2.5174, -0.9270, 2.2638 \cdot 10^{-6})$
	50	55	55	0.03	$(3.0000, -46.5000, 93.0000)$	53	0.02	$(8.4209, -32.8029, 18.7489)$
	100	77	77	0.38	$(\mathbf{3.8709}, \mathbf{-96.7675}, \mathbf{161.6056})^*$	67	0.07	$(7.2561, -92.4828, 178.8300)$
	250	44	26	0.10	$(14.8103, -235.1258, 286.1334)^*$	103	0.29	$(11.3995, -238.4833, 467.8050)$
	500	39	31	0.28	$(2.3476, -498.4372, 974.0640)^*$	187	1.51	$(8.2359, -492.4426, 962.2006)$

* Scaling procedure is utilized in MDBDC

** Locality measure is chosen to be 0.9 (default: 0.5) in MPB

In Table 4, the columns n_f and n_{ξ} for MDBDC contain also the evaluations used in Algorithm 3 guaranteeing Clarke stationarity. For example, in test problem 15 with $n = 10$ we have $n_f = 2225$ where 458 of them are obtained from Algorithm 3 and $n_{\xi} = 1980$, where 186 of them are caused by Algorithm 3. Hence, we might need a relatively high number of evaluations to be able to stop. Nevertheless, MDBDC uses significantly less CPU time in test problems 13–15 and 18 with $n \geq 250$.

To conclude, MDBDC performs well in the test problems reported. Indeed, in the small test problems ($2 \leq n \leq 100$), the average number of function calls is 277.97 and the average number of subgradient evaluations is 255.62 for MDBDC while the average number of evaluations for MPB is 305.43. In the larger test problems reported ($n > 100$), the average number of function calls is 255.80 and the average number of subgradient evaluations is 224.33 for MDBDC while the average number of evaluations for MPB is even 5277.87. Thus, MDBDC uses less evaluations on average in the test problems reported. Additionally, by utilizing some kind of aggregation in MDBDC, it might be possible to decrease the number of evaluations needed even more.

5 Conclusions

We have proposed a new descent method for the multiobjective DC optimization (MDBDC) producing weakly Pareto stationary solutions, and the method is proved to be finitely convergent under mild assumptions. This method can be used by executing it several times with different starting points to obtain an approximation of the set of local weak Pareto optima. Other possibility is to use MDBDC as a part of some interactive method like in [27, 28, 31]. Additionally, it can be used to solve single-objective DC problems with DC constraints to obtain a Clarke stationary solution.

The numerical experiments have shown the good performance of the method. The results obtained by comparing MDBDC and the multiobjective proximal bundle method [25, 28] validate the use of the method specially designed for multiobjective DC optimization instead of the method for general nonconvex multiobjective optimization. With more accurate model capturing the convex and the concave behaviour, we can learn more about the objectives and hence obtain better solutions. In future, the implementation of MDBDC could be improved by adding some sort of aggregation strategy [17, 18]. Moreover, MDBDC could be used in some practical applications, like data classification or cluster analysis.

Acknowledgements The research is financially supported by University of Turku Graduate School UTUGS Matti programme, and Academy of Finland project No. 289500.

The authors gratefully thank Marko Mäkelä and Napsu Karmitsa for their valuable comments and encouragement during the preparation of this paper.

References

[1] A. Astorino and G. Miglionico. Optimizing sensor cover energy via DC programming. *Optimization Letters*, 10(2):355–368, 2016.

[2] A. Bagirov, N. Karmitsa, and M. M. Mäkelä. *Introduction to Nonsmooth Optimization: Theory, Practice and Software*. Springer, Cham Heidelberg, 2014.

[3] A. Bagirov and J. Yearwood. A new nonsmooth optimization algorithm for minimum sum-of-squares clustering problems. *European Journal of Operational Research*, 170(2):578–596, 2006.

[4] J. Y. Bello Cruz and A. N. Iusem. A strongly convergent method for nonsmooth convex minimization in Hilbert spaces. *Numerical Functional Analysis and Optimization*, 32(10):1009–1018, 2011.

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

[6] F. H. Clarke. *Optimization and Nonsmooth Analysis*. John Wiley & Sons, Inc., New York, 1983.

[7] A. Fuduli, M. Gaudioso, and G. Giallombardo. A DC piecewise affine model and bundling technique in nonconvex nonsmooth minimization. *Optimization Methods and Software*, 19(1):89–102, 2004.

[8] N. Gadhi and A. Metrane. Sufficient optimality condition for vector optimization problems under d.c. data. *Journal of Global Optimization*, 28(1):55–66, 2004.

[9] M. Gaudioso, T. V. Gruzdeva, and A. S. Strekalovsky. On numerical solving the spherical separability problem. *Journal of Global Optimization*, 66(1):21–34, 2016.

[10] P. Hartman. On functions representable as a difference of convex functions. *Pacific Journal of Mathematics*, 9(3):707–713, 1959.

[11] J.-B. Hiriart-Urruty. Generalized differentiability, duality and optimization for problems dealing with differences of convex functions. *Lecture Note in Economics and Mathematical Systems*, 256:37–70, 1985.

[12] K. Holmberg and H. Tuy. A production-transportation problem with stochastic demand and concave production costs. *Mathematical Programming*, 85(1):157–179, 1999.

[13] R. Horst and N. V. Thoai. DC programming: Overview. *Journal of Optimization Theory and Applications*, 103(1):1–43, 1999.

[14] Y. Ji, M. Goh, and R. de Souza. Proximal point algorithms for multi-criteria optimization with the difference of convex objective functions. *Journal of Optimization Theory and Applications*, 169(1):280–289, 2016.

[15] K. Joki, A. Bagirov, N. Karmitsa, and M. M. Mäkelä. A proximal bundle method for nonsmooth DC optimization utilizing nonconvex cutting planes. *Journal of Global Optimization*, 2016.

[16] K. Joki, A. Bagirov, N. Karmitsa, M. M. Mäkelä, and S. Taheri. Double bundle method for nonsmooth DC optimization. Technical Report 1173, TUCS Technical Reports, Turku Centre for Computer Science, Turku, 2017.

[17] K. C. Kiwiel. An aggregate subgradient method for nonsmooth convex minimization. *Mathematical Programming*, 27(3):320–341, 1983.

[18] K. C. Kiwiel. A descent method for nonsmooth convex multiobjective minimization. *Large Scale Systems*, 8(2):119–129, 1985.

[19] K. C. Kiwiel. Proximity control in bundle methods for convex nondifferentiable optimization. *Mathematical Programming*, 46(1):105–122, 1990.

[20] H. A. Le Thi and T. Pham Dinh. Solving a class of linearly constrained indefinite quadratic problems by d.c. algorithms. *Journal of Global Optimization*, 11(3):253–285, 1997.

[21] H. A. Le Thi and T. Pham Dinh. The DC (difference of convex functions) programming and DCA revisited with DC models of real world nonconvex optimization problems. *Annals of Operations research*, 133(1):23–46, 2005.

[22] L. Lukšan. Dual method for solving a special problem of quadratic programming as a subproblem at linearly constrained nonlinear minimax approximation. *Kybernetika*, 20(6):445–457, 1984.

[23] M. M. Mäkelä. Multiobjective Proximal Bundle Method for Nonconvex Nonsmooth Optimization: Fortran Subroutine MPBNGC 2.0. Technical Report B 13/2003, Reports of the Department of Mathematical Information Technology, Series B, Scientific computing, University of Jyväskylä, Jyväskylä, 2003.

[24] M. M. Mäkelä, V.-P. Eronen, and N. Karmitsa. On Nonsmooth Multiobjective Optimality Conditions with Generalized Convexities. In Th. M. Rassias, C. A. Floudas, and S. Butenko, editors, *Optimization in Science and Engineering*, pages 333–357. Springer, 2014.

[25] M. M. Mäkelä, N. Karmitsa, and O. Wilppu. Proximal Bundle Method for Nonsmooth and Nonconvex Multiobjective Optimization. In T. Tuovinen,

S. Repin, and P. Neittaanmäki, editors, *Mathematical Modeling and Optimization of Complex Structures*, volume 40 of *Computational Methods in Applied Sciences*, pages 191–204. Springer, 2016.

[26] M. M. Mäkelä and P. Neittaanmäki. *Nonsmooth Optimization: Analysis and Algorithms with Applications to Optimal Control*. World Scientific Publishing Co., Singapore, 1992.

[27] K. Miettinen. *Nonlinear Multiobjective Optimization*. Kluwer Academic Publishers, Boston, 1999.

[28] K. Miettinen and M. M. Mäkelä. Interactive bundle-based method for nondifferentiable multiobjective optimization: NIMBUS. *Optimization*, 34(3):231–246, 1995.

[29] E. S. Mistakidis and G. E. Stavroulakis. *Nonconvex Optimization in Mechanics. Smooth and Nonsmooth Algorithms, Heuristics and Engineering Applications by the F.E.M.* Kluwer Academic Publisher, Dordrecht, 1998.

[30] J. J. Moreau, P. D. Panagiotopoulos, and G. Strang (Eds.). *Topics in Nonsmooth Mechanics*. Birkhäuser, Basel, 1988.

[31] H. Mukai. Algorithms for Multicriterion Optimization. *IEEE Transactions on Automatic Control*, ac-25(2):177–186, 1979.

[32] J. Outrata, M. Koĉvara, and J. Zowe. *Nonsmooth Approach to Optimization Problems with Equilibrium Constraints. Theory, Applications and Numerical Results*. Kluwer Academic Publishers, Dordrecht, 1998.

[33] T. Pham Dinh and H. A. Le Thi. Convex analysis approach to DC programming: Theory, algorithms and applications. *Acta Mathematica Vietnamica*, 22(1):289–355, 1997.

[34] S. Qu, M. Goh, S. Y. Wu, and R. De Souza. Multiobjective DC programs with infinite convex constraints. *Journal of Global Optimization*, 59(1):41–58, 2014.

[35] S. Qu, C Liu, M. Goh, Y. Li, and Y. Ji. Nonsmooth multiobjective programming with quasi-Newton methods. *European Journal of Operational Research*, 235(3):503–510, 2014.

[36] H. Schramm and J. Zowe. A version of the bundle idea for minimizing a nonsmooth function: Conceptual idea, convergence analysis, numerical results. *SIAM Journal on Optimization*, 2(1):121–152, 1992.

[37] W. Y. Sun, R. J. B. Sampaio, and M. A. B. Candido. Proximal point algorithm for minimization of DC functions. *Journal of Computational Mathematics*, 21(4):451–462, 1979.

[38] A. Taa. Optimality conditions for vector optimization problems of a difference of convex mappings. *Journal of Global Optimization*, 31(3):421–436, 2005.

[39] J. F. Toland. On subdifferential calculus and duality in nonconvex optimization. *Mémoires de la Société Mathématique de France*, 60:173–180, 1979.

[40] S. Wang. Algorithms for Multiobjective and Nonsmooth Optimization. In P. Kleinschmidt, F.J. Radermacher, W. Sweitzer, and H. Wildermann, editors, *Methods of Operations Research, 58*, pages 131–142. Athenaum Verlag, 1989.

TURKU

CENTRE *for*

COMPUTER

SCIENCE

Joukahaisenkatu 3-5 A, 20520 TURKU, Finland | www.tucs.fi

University of Turku

Faculty of Mathematics and Natural Sciences

- Department of Information Technology
- Department of Mathematics and Statistics

Turku School of Economics

- Institute of Information Systems Sciences

Åbo Akademi University

- Computer Science
- Computer Engineering

ISBN 978-952-12-3501-6
ISSN 1239-1891