Outi Wilppu | Napsu Karmitsa | Marko M. Mäkelä

New Multiple Subgradient Descent Bundle Method for Nonsmooth Multiobjective Optimization

Turku Centre for Computer Science

TUCS Technical Report
No 1126, December 2014

New Multiple Subgradient Descent Bundle Method for Nonsmooth Multiobjective Optimization

Outi Wilppu
University of Turku, Department of Mathematics and Statistics
FI-20014 Turku, Finland
outi.wilppu@utu.fi
Napsu Karmitsa
University of Turku, Department of Mathematics and Statistics
FI-20014 Turku, Finland
napsu@karmitsa.fi
Marko M. Mäkelä
University of Turku, Department of Mathematics and Statistics
FI-20014 Turku, Finland
makela@utu.fi

TUCS Technical Report
No 1126, December 2014

Abstract

The aim of this paper is to propose a new multiple subgradient descent bundle method for solving unconstrained convex nonsmooth multiobjective optimization problems. Contrary to many existing multiobjective optimization methods, our method treats the objective functions as they are without employing any scalarization. The main idea is to find descent directions for every objective function separately and then form a common descent direction for every objective function. In addition, we prove that the method is convergent and it finds weakly Pareto optimal solutions. Finally, some numerical experiments are considered.

Keywords: Bundle methods, Descent methods, Multiobjective optimization, Nonsmooth optimization

TUCS Laboratory
Turku Optimization Group (TOpGroup)

1 Introduction

Multiobjective nature arises in many practical applications. There are several conflicting objectives to be optimized and the aim is to find a compromise between these different goals being as good as possible for all the objectives at the same time. The compromise is optimal if we cannot improve any objective without deteriorating some other. The problems of this kind are called multiobjective optimization problems and these problems exist in many areas, for example, in engineering [23], economics [26] and mechanics [24].

The most of the existing multiobjective optimization methods convert the multiple objectives to the single-objective problem and apply some single-objective method to solve it. This process is called a scalarization (see e.g. [7, 21]). In this paper, instead of the scalarization, we are focusing on descent methods employing the objectives as they are. The method is said to be descent if at every iteration it moves to the direction where the values of all objectives improve and the new iteration points produced give better values for objective functions. For differentiable functions there are several methods of this type described in literature e.g. [6, 8, 9, 25].

In addition to multiobjective characteristic, many of the real-life problems have also nonsmooth nature meaning that the functions are not necessarily differentiable in classical sense. There are several reasons for the nonsmoothness of objective functions [2, 17]: first, an objective function itself can be nondifferentiable as are, for example, piecewise linear tax models in economics [14]; second, some technical constraints may cause a nonsmooth dependence between the variables and the functions even the objective functions are continuously differentiable as in, for instance, obstacle problems in optimal shape design [10]; third, some optimization methods for a constrained problem may lead to a nonsmooth problem as, for example, the exact penalty function method [2]; and fourth, the problem may be analytically smooth but numerically behave like a nonsmooth problem. Since nonsmooth objective functions are not differentiable we cannot utilize gradient-based methods to solve the problems.

Bundle methods [2, 11, 12, 13, 17, 20] are considered as the most efficient way to solve nonsmooth single-objective optimization problems. The idea in these methods is to approximate the subdifferential (i.e. a set of generalized gradients, the so-called subgradients [4]) of the objective function with a bundle including information from the neighborhood of the iteration point. The only assumptions needed are that one arbitrary subgradient and the value of the objective function can be evaluated at every point. In order to improve a general bundle method for single-objective optimization to a more efficient one the idea of the proximal bundle method [13, 20] can be utilized. In this method a weighting parameter has been added to the stabilizing term. The stabilizing term makes sure that the approximation of the function is close enough to the iteration point and it also guarantees the existences of the search directions.

As noted before, there exists various methods either for smooth (i.e. continuously differentiable) multiobjective optimization or for nonsmooth single-objective optimization but only few methods are designed for nonsmooth multiobjective optimization. Since many multiobjective problems has nonsmooth objectives, we study here nonsmooth multiobjective optimization. We are focusing on descent methods employing the objectives as they are. In literature methods of this kind are described e.g. in [11, 19, 22, 27] where these methods utilize the basic bundle idea. This type of the methods may be applied, for example, in interactive methods proposed in [22, 25].

In this paper we propose a new multiple subgradient descent bundle method (MSGDB) for convex nonsmooth multiobjective optimization problems. MSGDB generalizes the ideas of smooth multiple-gradient descent algorithm [5, 6] which, in its turn, extends the well-known steepest descent method for the multiobjective problems. The nonsmoothness of the objectives is taken into account by using the proximal bundle idea. That is, in MSGDB, all the objective functions are first linearized separately and the proximal bundle approach is used to find subgradients giving descent directions for each objective function. After that a convex hull of the subgradients is formed and a minimum norm element is calculated to obtain a common descent direction for all the objective functions.

We also recall the basic idea of the multiobjective proximal bundle method (MPB) [19, 22] which is used as a reference method. In the numerical experiments, MSGDB is compared with MPB. Both of these methods form a common descent direction for every objective function and they are based on the proximal bundle approach. However, they utilize the idea of the proximal bundle in different way. MPB is a generalization of the proximal bundle method utilizing an improvement function taking all the objectives into account at the same time. The improvement function is then linearized in order to obtain the descent search direction [19, 22].

The paper is organized as follows. In Section 2, we recall some basic results from multiobjective and nonsmooth optimization. Section 3 is devoted to the methods and we describe the new MSGDB method and for the sake of comparison we recall the basic ideas of MPB. Some numerical experiments are given in Section 4. In conclusion, Section 5, we give some final remarks.

2 Preliminaries

Let us consider an unconstrained optimization problem of the form

$$\begin{aligned} \min \quad & \{f_1(\boldsymbol{x}), \ldots, f_m(\boldsymbol{x})\} \\ \text{s.t.} \quad & \boldsymbol{x} \in \mathbb{R}^n, \end{aligned} \tag{1}$$

where "min" means that all the objective functions are minimized simultaneously. The objective functions $f_i : \mathbb{R}^n \to \mathbb{R}$, $i = 1, \ldots, m$ are assumed to be convex but not necessarily differentiable. Since the objective functions are convex they are known to be also *locally Lipschitz continuous* [2]. The function f_i is locally Lipschitz continuous at the point $\boldsymbol{x} \in \mathbb{R}^n$ if there exist $K > 0$ and $\varepsilon > 0$ such that

$$|f_i(\boldsymbol{y}) - f_i(\boldsymbol{z})| \leq K\|\boldsymbol{y} - \boldsymbol{z}\| \text{ for all } \boldsymbol{y}, \boldsymbol{z} \in B(\boldsymbol{x}; \varepsilon),$$

where $B(\boldsymbol{x}; \varepsilon)$ is an open ball with center $\boldsymbol{x}$ and radius ε.

At first we recall some basic results from multiobjective and nonsmooth optimization. For more details we refer to [2, 4, 7, 20, 21]. Notation $\boldsymbol{x} < \boldsymbol{y}$ is used if $x_i < y_i$ for all $i \in \{1, \ldots, n\}$ and $\boldsymbol{x} \leq \boldsymbol{y}$ if $x_i \leq y_i$ for all $i \in \{i, \ldots, n\}$.

A solution $\boldsymbol{x}^* \in \mathbb{R}^n$ of the problem (1) is called *Pareto optimal* if there does not exist another point $\boldsymbol{x} \in \mathbb{R}^n$ such that $f_i(\boldsymbol{x}) \leq f_i(\boldsymbol{x}^*)$ for all $i = 1, \ldots, m$ and $f_j(\boldsymbol{x}) < f_j(\boldsymbol{x}^*)$ for at least one index $j \in \{1, \ldots, m\}$. This definition means that no objective can be improved without impairing some other objective at the same time. Usually there exist several Pareto optimal solutions being all mathematically equally good.

A generalized concept, called *weak Pareto optimality*, is also possible to define. A solution $\boldsymbol{x}^* \in \mathbb{R}^n$ of the problem (1) is weakly Pareto optimal if there does not exist another point $\boldsymbol{x} \in \mathbb{R}^n$ such that $f_i(\boldsymbol{x}) < f_i(\boldsymbol{x}^*)$ for all $i = 1, \ldots, m$. This means that there does not exist any other point such that all objective functions f_i have better values. Clearly every Pareto optimal solution is also weakly Pareto optimal.

Nonsmooth functions do not have gradient at every point and thus instead of classical gradient a generalized gradient called subgradient need to be utilized. If function $f_i : \mathbb{R}^n \to \mathbb{R}$ is convex, then the *subdifferential* of the function f_i at the point $\boldsymbol{x}$ is a set

$$\partial f_i(\boldsymbol{x}) = \left\{\boldsymbol{\xi}_i \in \mathbb{R}^n \mid f_i(\boldsymbol{y}) \geq f_i(\boldsymbol{x}) + \boldsymbol{\xi}_i^T(\boldsymbol{y} - \boldsymbol{x}) \text{ for all } \boldsymbol{y} \in \mathbb{R}^n\right\}.$$

A vector $\boldsymbol{\xi}_i \in \partial f_i(\boldsymbol{x})$ is called a *subgradient* of the function f_i at the point $\boldsymbol{x}$. Later we assume that at least one arbitrary subgradient can be evaluated at every point for every objective function.

The subgradient of a differentiable convex function is unique and equals to its gradient as stated in the following theorem.

Theorem 2.1. [2, 20] *Let function $f_i : \mathbb{R}^n \to \mathbb{R}$ be convex and differentiable at the point $\boldsymbol{x}$. Then*

$$\partial f_i(\boldsymbol{x}) = \{\nabla f_i(\boldsymbol{x})\}.$$

The optimization method is called descent if at every iteration it produces a better solution for the problem (1). In order to find this better solution the concept of a descent direction is needed. The direction $\boldsymbol{d} \in \mathbb{R}^n$ is said to be a *descent direction* for function $f_i : \mathbb{R}^n \to \mathbb{R}$ at $\boldsymbol{x}$ if there exists $\varepsilon > 0$ such that

$$f_i(\boldsymbol{x} + t\boldsymbol{d}) < f_i(\boldsymbol{x}) \quad \text{for all } t \in (0, \varepsilon].$$

The next theorem shows how descent directions can be found.

Theorem 2.2. [2, 20] *Let function* $f_i : \mathbb{R}^n \to \mathbb{R}$ *be convex. The direction* $\boldsymbol{d} \in \mathbb{R}^n$ *is a descent direction for* f_i *at the point* $\boldsymbol{x}$ *if*

$$\boldsymbol{\xi}_i^T \boldsymbol{d} < 0 \quad \text{for all } \boldsymbol{\xi}_i \in \partial f_i(\boldsymbol{x}).$$

A well-known necessary and sufficient condition for global optimality in convex single-objective case is the following

Theorem 2.3. [2, 20] *A convex function* $f_i : \mathbb{R}^n \to \mathbb{R}$ *attains its global minimum at the point* $\boldsymbol{x}$ *if and only if*

$$\mathbf{0} \in \partial f_i(\boldsymbol{x}).$$

The following theorem is the multiobjective counterpart for Theorem 2.3 giving us necessary and sufficient conditions for (weak) Pareto optimality.

Theorem 2.4. [21] *Consider the problem* (1). *A necessary condition for a point* $\boldsymbol{x}$ *to be a (weakly) Pareto optimal solution of the problem* (1) *is that there exist multipliers* $\boldsymbol{\lambda} \in \mathbb{R}^m$, $\boldsymbol{\lambda} \geq \mathbf{0}$ *and* $\boldsymbol{\lambda} \neq \mathbf{0}$ *such that*

$$\mathbf{0} \in \sum_{i=1}^{m} \lambda_i \partial f_i(\boldsymbol{x}).$$

The above mentioned condition is also sufficient for weak Pareto optimality and for Pareto optimality if $\boldsymbol{\lambda} > \mathbf{0}$.

3 Methods

In this section, two different descent methods for multiobjective optimization utilizing the proximal bundle idea are described. The first one is a new MSGDB for unconstrained convex multiobjective optimization problems. In addition, we recall ideas of MPB and use it as a reference method in order to make comparisons between these two methods.

3.1 Multiple subgradient descent bundle method

MSGDB extends the ideas of the multiple-gradient descent algorithm [5, 6]. This method, in its turn, is a generalization of the classical steepest descent method for smooth multiobjective optimization. In the multiple-gradient descent algorithm the direction of the negative gradient is utilized. This direction is known to be a descent direction for a smooth function [3, 20]. By knowing the gradients giving descent directions for every objective separately, a convex hull of these gradients is formulated. Then the minimum norm element of that convex hull can be found. The negative direction of this minimum norm element gives a common descent direction for all the objective functions.

In the following, we assume that the objective functions are nonsmooth and thus gradients cannot be employed. Instead of gradients, we utilize subgradients in MSGDB. However, the gradients cannot be just replaced with subgradients since there is no guarantee that the opposite direction of an arbitrary subgradient would be a descent direction [20]. If the whole subdifferential of the objective function was known, the steepest descent direction could be calculated but this is, in most cases, too demanding requirement in practice. That is why the bundle approach is utilized in order to find a descent direction.

The basic idea behind the bundle method is to approximate the whole subdifferential of the objective function by gathering information from the neighborhood of the iteration point into the bundle. Thus only one arbitrary subgradient from the subdifferential and the value of the function at each iteration point need to be evaluated. We refer to [2, 11, 12, 13, 17, 20] for more details about bundle methods.

Next we present how to calculate descent directions for an individual objective f_i, $i = 1, \dots, m$. These calculations are performed for every objective separately. Consider an iteration point $\boldsymbol{x}_k$ at iteration k and some auxiliary points $\boldsymbol{y}_{i,j}$, $j \in J_{i,k}$ from past iterations, where $J_{i,k}$ is a nonempty subset of $\{1, \dots, k\}$. In addition, some arbitrary subgradients $\boldsymbol{\xi}_{i,j} \in \partial f_i(\boldsymbol{y}_{i,j})$ for $i = 1, \dots, m$ and $j \in J_{i,k}$ are supposed to be known.

The following piecewise linear model also called a *cutting plane model* is formed to approximate the function f_i

$$\hat{f}_i^k(\boldsymbol{x}) = \max_{j \in J_{i,k}} \left\{ f_i(\boldsymbol{x}_k) + \boldsymbol{\xi}_{i,j}^T(\boldsymbol{x} - \boldsymbol{x}_k) - \alpha_{i,j}^k \right\} \text{ for all } i = 1, \dots, m, \tag{2}$$

where the linearization error $\alpha_{i,j}^k$ is defined by

$$\alpha_{i,j}^k = f_i(\boldsymbol{x}_k) - f_i(\boldsymbol{y}_{i,j}) - \boldsymbol{\xi}_{i,j}^T(\boldsymbol{x}_k - \boldsymbol{y}_{i,j}) \text{ for all } j \in J_{i,k}. \tag{3}$$

The search direction $\boldsymbol{d}_{i,k}$ can then be calculated from formula

$$\boldsymbol{d}_{i,k} = \operatorname*{argmin}_{\boldsymbol{d}_i \in \mathbb{R}^n} \left\{ \hat{f}_i^k(\boldsymbol{x}_k + \boldsymbol{d}_i) + \frac{1}{2} u_{i,k} \left\| \boldsymbol{d}_i \right\|^2 \right\} \text{ for all } i = 1, \dots, m, \tag{4}$$

where $u_{i,k}$ is a positive weighting parameter. The term $\frac{1}{2} u_{i,k} \left\| \boldsymbol{d}_i \right\|^2$ is a stabilizing term guaranteeing the existence and the uniqueness of the solution and keeping the approximation local enough.

It is possible to rewrite the nonsmooth problem (4) for each objective functions f_i in the following smooth form

$$\begin{aligned} \min_{\boldsymbol{d}_i \in \mathbb{R}^n, v_i \in \mathbb{R}} \quad & v_i + \frac{1}{2} u_{i,k} \left\| \boldsymbol{d}_i \right\|^2 \\ \text{s.t.} \quad & -\alpha_{i,j}^k + \boldsymbol{\xi}_{i,j}^T \boldsymbol{d}_i \le v_i \quad \text{for all } j \in J_{i,k}. \end{aligned} \tag{5}$$

The problem (5) can be made easier and instead of the problem (5) we can solve its quadratic dual problem

$$\begin{aligned}\min_{\lambda_i \in \mathbb{R}} \quad & \frac{1}{2u_{i,k}} \sum_{j \in J_{i,k}} \lambda_{i,j} \boldsymbol{\xi}_{i,j} + \sum_{j \in J_{i,k}} \lambda_{i,j} \alpha_{i,j}^k \\ \text{s.t.} \quad & \sum_{j \in J_{i,k}} \lambda_{i,j} = 1 \\ & \lambda_{i,j} \geq 0.\end{aligned}$$

A unique solution of the problem (5) is then of the form [20]

$$\begin{aligned}\boldsymbol{d}_{i,k} &= -\frac{1}{u_{i,k}} \sum_{j \in J_{i,k}} \lambda_{i,j} \boldsymbol{\xi}_{i,j} \\ v_{i,k} &= -\Big(\frac{1}{u_{i,k}} \Big\| \sum_{j \in J_{i,k}} \lambda_{i,j} \boldsymbol{\xi}_{i,j} \Big\|^2 + \sum_{j \in J_{i,k}} \lambda_{i,j} \alpha_{i,j}^k \Big).\end{aligned}$$

Next a new auxiliary point $\boldsymbol{y}_{i,k+1} = \boldsymbol{x}_k + \boldsymbol{d}_{i,k}$ and the function value $f_i(\boldsymbol{y}_{i,k+1})$ is calculated. The procedure can be stopped and set $\boldsymbol{d}_{i,k}^* = \boldsymbol{d}_{i,k}$ if

$$f_i(\boldsymbol{y}_{i,k+1}) \leq f_i(\boldsymbol{x}_k) + m_L v_{i,k}, \tag{6}$$

where $m_L \in (0, \frac{1}{2})$ is a line search parameter. Note that $v_{i,k}$ has the following form [20]

$$v_{i,k} = \hat{f}_i^k(\boldsymbol{y}_{i,k+1}) - f_i(\boldsymbol{x}_k)$$

being a predicted descent of the function f_i at the point $\boldsymbol{x}_k$. This implies that the obtained function value at the new iteration point is significantly better than the function value at the previous iteration point. In bundle methods, if the condition (6) holds, a new iteration point $\boldsymbol{x}_{k+1}$ is calculated and this step is called a *serious step*. If the condition (6) does not hold we perform a *null step*, where the model will be improved by adding new information to the bundle. This is done by updating the bundle such that a new index is added to the set $J_{i,k+1} = J_{i,k} \bigcup \{k+1\}$. In addition, the subgradient $\boldsymbol{\xi}_{i,k+1} \in \partial f_i(\boldsymbol{y}_{i,k+1})$, the trial point $\boldsymbol{y}_{i,k+1}$ and the function value $f_i(\boldsymbol{y}_{i,k+1})$ are added to the bundle. The iteration point is updated by setting $\boldsymbol{x}_{k+1} = \boldsymbol{x}_k$. After that, a new value for the direction $\boldsymbol{d}_{i,k+1}$ can be calculated.

Null steps are continued until the condition (6) is satisfied and the sufficient descent is reached. It can be proved that the number of null steps is finite until the sufficient descent is reached [12].

According to Theorem 5.2.8 in [20] for the solution $\boldsymbol{d}_{i,k}$ of the problem (4) holds that $-\boldsymbol{d}_{i,k} \in \partial f_i(\boldsymbol{x}_k)$ and notation

$$\boldsymbol{d}_{i,k} = -\boldsymbol{\xi}_{i,k}^*, \text{ where } \boldsymbol{\xi}_{i,k}^* \in \partial f_i(\boldsymbol{x}_k)$$

can be used. Theorem 5.2.8 in [20] also shows that the direction obtained is descent for the estimated function $\hat{f}_i^k$. The following theorem shows that the direction obtained is descent also for the original objective function f_i.

Theorem 3.1. [20] *Let a function $f_i : \mathbb{R}^n \to \mathbb{R}$ be locally Lipschitz continuous at the point $\boldsymbol{x}$. The direction $\boldsymbol{d} \in \mathbb{R}^n$ is a descent direction for the function f_i at the point $\boldsymbol{x}$ if the direction $\boldsymbol{d}$ is a descent direction for the estimated function $\hat{f}_i$ at the point $\boldsymbol{x}$.*

Above we have shown how to calculate descent directions for an individual objective functions f_i, $i = 1, \dots, m$. Next the calculation of a common descent direction for all the objective functions is considered.

From Theorem 2.4 the following definition of Pareto stationarity is obtained to get a generalization of the Pareto optimality.

Definition 3.2. A point $\boldsymbol{x}$ is said to be *Pareto stationary* if there exist subgradients $\boldsymbol{\xi}_i \in \partial f_i(\boldsymbol{x})$ and multipliers $\lambda_i \geq 0$ for all $i = 1, \dots, m$, $\sum_{i=1}^m \lambda_i = 1$ such that

$$\sum_{i=1}^{m} \lambda_i \boldsymbol{\xi}_i = \boldsymbol{0}.$$

Note that due to Theorem 2.4 in the convex case Pareto stationarity equals to weak Pareto optimality.

In [6] Lemma 2.1 and Theorem 2.2 were proven to guarantee the functionality of the multiple-gradient descent algorithm. Same kind of results can be formulated also for MSGDB as will be shown in Lemma 3.3 and Theorem 3.4.

Lemma 3.3. *Let $\boldsymbol{d}_i = -\boldsymbol{\xi}_i^*$ with $\boldsymbol{\xi}_i \in \partial f_i(\boldsymbol{x})$ be a descent direction for f_i at the point $\boldsymbol{x}$ for all $i = 1, \dots, m$. Let C be a set of convex combinations of corresponding subgradients, that is,*

$$C = \text{conv}\{\boldsymbol{\xi}_i^* \mid i = 1, \dots, m\}, \tag{7}$$

where conv *denotes the convex hull of a set. Then there exists a unique vector $\boldsymbol{p}^* = \operatorname{argmin}_{\boldsymbol{p} \in C} \|\boldsymbol{p}\|$ such that*

$$\boldsymbol{p}^T \boldsymbol{p}^* \geq \boldsymbol{p}^{*T} \boldsymbol{p}^* = \|\boldsymbol{p}^*\|^2 \text{ for all } \boldsymbol{p} \in C. \tag{8}$$

Proof. If $\boldsymbol{0} \in C$, then it is a minimum norm element, $\boldsymbol{x}$ is Pareto stationary and the statement (8) is trivially valid. Assume that $\boldsymbol{0} \notin C$. Since C is a nonempty, closed and convex set, there exists a unique minimum norm element $\boldsymbol{p}^* \in C$ according to the closest point theorem [3].

Let a vector $\boldsymbol{p}$ be an arbitrary element of C and set $\boldsymbol{r} = \boldsymbol{p} - \boldsymbol{p}^*$. Due to convexity of C we have

$$\lambda \boldsymbol{r} + \boldsymbol{p}^* = \lambda \boldsymbol{p} + (1 - \lambda)\boldsymbol{p}^* \in C \quad \text{for all} \ \ \lambda \in [0, 1].$$

Since $\boldsymbol{p}^*$ is the minimum norm element, we have $\|\lambda \boldsymbol{r} + \boldsymbol{p}^*\| \geq \|\boldsymbol{p}^*\|$ implying

$$\|\lambda \boldsymbol{r} + \boldsymbol{p}^*\|^2 - \|\boldsymbol{p}^*\|^2 = 2\lambda \boldsymbol{r}^T \boldsymbol{p}^* + \lambda^2 \boldsymbol{r}^T \boldsymbol{r} \geq 0.$$

Since λ can be arbitrary small we get

$$0 \leq \boldsymbol{r}^T \boldsymbol{p}^* = (\boldsymbol{p} - \boldsymbol{p}^*)^T \boldsymbol{p}^* = \boldsymbol{p}^T \boldsymbol{p}^* - \boldsymbol{p}^{*T} \boldsymbol{p}^*$$

implying the statement (8). □

Combining the information from Definition 3.2 and Lemma 3.3 the following theorem is obtained.

Theorem 3.4. *Let $\boldsymbol{d}_i = -\boldsymbol{\xi}_i^*$ with $\boldsymbol{\xi}_i \in \partial f_i(\boldsymbol{x})$ be a descent direction for f_i at the point $\boldsymbol{x}$ for all $i = 1, \ldots, m$. Let C be as in* (7) *and C° be a set of strictly convex combinations of subgradients $\boldsymbol{\xi}_i^*$, in other words*

$$C^\circ = \text{int conv}\{\boldsymbol{\xi}_i^* \mid i = 1, \ldots, m\}.$$

If a vector $\boldsymbol{d}^*$ is of form $\boldsymbol{d}^* = -\boldsymbol{p}^*$, where $\boldsymbol{p}^* = \operatorname{argmin}_{\boldsymbol{p} \in C} \|\boldsymbol{p}\|$, then either we have

1. *$\boldsymbol{d}^* = \boldsymbol{0}$ and the point $\boldsymbol{x}$ is Pareto stationary.*

or

2. *$\boldsymbol{d}^* \neq \boldsymbol{0}$ and the vector $\boldsymbol{d}^*$ is a common descent direction for every objective function. Moreover, if $\boldsymbol{p}^* \in C^\circ$ then $\boldsymbol{p}^T \boldsymbol{p}^* = \|\boldsymbol{p}^*\|^2$ for all $\boldsymbol{p} \in C$.*

Proof. Consider the first case. Now the vector $-\boldsymbol{d}^* = \boldsymbol{p}^* = \sum_{i=1}^m \lambda_i^* \boldsymbol{\xi}_i^* = \boldsymbol{0}$, $\lambda_i^* \geq 0$ for all $i = 1, \ldots, m$ and $\sum_{i=1}^m \lambda_i^* = 1$. Thus the point $\boldsymbol{x}$ is Pareto stationary.

Consider then the second case. Now the vector $-\boldsymbol{d}^* = \boldsymbol{p}^* = \sum_{i=1}^m \lambda_i^* \boldsymbol{\xi}_i^* \neq \boldsymbol{0}$ and thus the point $\boldsymbol{x}$ is not Pareto stationary. The vector $\boldsymbol{p}^*$ is assumed to be the minimum norm element of the set C. Since also $\boldsymbol{\xi}_i^* \in C$ for all i we have $\boldsymbol{\xi}_i^{*T} \boldsymbol{p}^* \geq \|\boldsymbol{p}^*\|^2 > 0$ according to Lemma 3.3 and thus for the direction $\boldsymbol{d}^*$ it holds $\boldsymbol{\xi}_i^{*T}(\boldsymbol{d}^*) = -\boldsymbol{\xi}_i^{*T} \boldsymbol{p}^* < 0$. Then according to Theorem 2.2 the direction $\boldsymbol{d}^*$ is a descent direction for all functions f_i with $i = 1, \ldots, m$.

Next we prove that if the vector $\boldsymbol{p}^* \in C^\circ$ then $\boldsymbol{p}^T \boldsymbol{p}^* = \|\boldsymbol{p}^*\|^2$ for all $\boldsymbol{p} \in C$ of form $\boldsymbol{p} = \sum_{i=1}^m \alpha_i \boldsymbol{\xi}_i^*$, $\alpha_i \geq 0$ for all $i = 1, \ldots, m$ and $\sum_{i=1}^m \alpha_i = 1$. With assumptions of the theorem, the element $\boldsymbol{p}^*$ is a solution of the problem

$$\begin{aligned} \min \quad & \boldsymbol{p}^T \boldsymbol{p} \\ \text{s. t.} \quad & \sum_{i=1}^m \alpha_i = 1. \end{aligned} \tag{9}$$

Thus by using a vector $\boldsymbol{\alpha} \in \mathbb{R}^m$ the Lagrangian of (9) obtained is

$$L(\boldsymbol{\alpha}, \lambda) = \boldsymbol{p}^T \boldsymbol{p} + \lambda(\sum_{i=1}^{m} \alpha_i - 1)$$

and the vector $\boldsymbol{\alpha}$ satisfies the following optimality conditions in optimum

$$\frac{d\,L(\boldsymbol{\alpha}^*, \lambda^*)}{d\,\alpha_i} = 0 \ \text{ for all } \ i, \ \text{ and } \ \frac{d\,L(\boldsymbol{\alpha}^*, \lambda^*)}{d\,\lambda} = 0.$$

The first condition implies that for every index i the following holds:

$$\frac{d\,(\boldsymbol{p}^T \boldsymbol{p})}{d\,\alpha_i} + \lambda = 0. \tag{10}$$

When $\boldsymbol{p} = \sum_{i=1}^{m} \alpha_i^* \boldsymbol{\xi}_i^*$, from the equation (10) it follows

$$\frac{d\,(\boldsymbol{p}^T \boldsymbol{p})}{d\,\alpha_i} = 2\Big(\frac{d\,\boldsymbol{p}}{d\,\alpha_i}\Big)^T \boldsymbol{p} = 2(\boldsymbol{\xi}_i^*)^T \boldsymbol{p}^* = -\lambda$$

and this equation implies that $\boldsymbol{\xi}_i^{*T} \boldsymbol{p}^* = -\frac{\lambda}{2}$ when $\alpha_i > 0$ for every i.

Consider an arbitrary element $\boldsymbol{p} \in C$ such that $\boldsymbol{p} = \sum_{i=1}^{m} \mu_i \boldsymbol{\xi}_i^*$, where $\mu_i \geq 0$ for all i and $\sum_{i=1}^{m} \mu_i = 1$. Now

$$\boldsymbol{p}^T \boldsymbol{p}^* = \sum_{i=1}^{m} \mu_i \boldsymbol{\xi}_i^{*T} \boldsymbol{p}^* = -\sum_{i=1}^{m} \mu_i \frac{\lambda}{2} = -\frac{\lambda}{2}.$$

On the other hand, we can choose $\boldsymbol{p} = \boldsymbol{p}^*$ and thus

$$\|\boldsymbol{p}^*\|^2 = \boldsymbol{p}^{*T} \boldsymbol{p}^* = \boldsymbol{p}^T \boldsymbol{p}^* = -\frac{\lambda}{2}.$$

□

The main result of Theorem 3.4 is that the direction $\boldsymbol{d}^* = -\boldsymbol{p}^* = -\sum_{i=1}^{m} \lambda_i^* \boldsymbol{\xi}_i^*$ is a common descent direction and it can be calculated by solving the problem

$$\begin{aligned} \min \quad & \Big\| \sum_{i=1}^{m} \lambda_i \boldsymbol{\xi}_i^* \Big\|^2 \\ \text{s.t.} \quad & \sum_{i=1}^{m} \lambda_i = 1 \\ & \lambda_i \geq 0, \quad \text{for all } \ i, \end{aligned} \tag{11}$$

which has a unique solution since the objective function of (11) is strictly convex.

We have now presented a method to calculate a common descent direction for all the objective functions. A stepsize t can then be calculated as is done in the multiple-gradient descent algorithm [6] by formulating functions $g_i : \mathbb{R} \to \mathbb{R}$ for all $i = 1, \ldots, m$ of the form

$$g_i(t_i) = f_i(\boldsymbol{x}_k + t_i \boldsymbol{d}^*) \text{ for all } i. \tag{12}$$

We apply some line search method in order to find intervals $[0, t_i]$, where the functions g_i are decreasing. By combining this information, an interval where all the functions g_i are decreasing can be obtained. The end point of this interval is the stepsize t.

Now we can describe an algorithm for MSGDB. The flow chart of the algorithm is presented in Figure 1.

Algorithm 1. Multiple subgradient descent bundle method (MSGDB)

Step 1: (*Initialization*) Select the starting point $\boldsymbol{x}_1$, the line search parameter $m_L \in (0, \frac{1}{2})$ and the stopping parameter $\varepsilon > 0$. Set an outer iteration index $l = 1$.

Step 2: (*Direction finding*) Do the following steps for all $i = 1, \ldots, m$ to calculate directions $\boldsymbol{d}_i$.

- Step A: (*Initialization*) Select the weighting parameter $u_{i,1}$. Set auxiliary point $\boldsymbol{y}_{i,1} = \boldsymbol{x}_1$ and a set $J_{i,1} = \{1\}$. Set also an inner iteration index $k = 1$.
- Step B: (*Direction finding*) Calculate a direction $\boldsymbol{d}_{i,k}$ from formula (4) and set $\boldsymbol{y}_{i,k+1} = \boldsymbol{x}_k + \boldsymbol{d}_{i,k}$. If the condition (6) holds, then set $\boldsymbol{d}_i = \boldsymbol{d}_{i,k}$. Otherwise go to step C.
- Step C: (*Update*) Set $J_{i,k+1} = J_{i,k} \bigcup \{k+1\}$, calculate $\boldsymbol{\xi}_{i,k+1} \in \partial f_i(\boldsymbol{y}_{i,k+1})$ and update $u_{i,k+1}$. Go to step B.

Step 3: (*Common descent direction finding*) Calculate a minimum norm element $\boldsymbol{p}^*$ of the set C (see (7)) by solving the problem (11). Set $\boldsymbol{d}_l = -\boldsymbol{p}^*$.

Step 4: (*Stopping criterion*) If $\|\boldsymbol{d}_l\| < \varepsilon$, then stop.

Step 5: (*Line search*) Calculate a stepsize t being the largest strictly positive real number for which all functions g_i (see (12)) are decreasing. Set $\boldsymbol{x}_{l+1} = \boldsymbol{x}_l + t\boldsymbol{d}_l$ and go to step 2.

In practice the size of the bundle need to be limited. The easiest way to do this is to choose some maximal size for the bundle, for example $J_{max} = n + 3$. The set $J_{i,k+1}$ is updated as in Step C if $|J_{i,k}| < J_{max}$ and if $|J_{i,k}| = J_{max}$ a set

$$J_{i,k+1} = J_{i,k} \bigcup \{k+1\} \setminus \{k - J_{max}\} \tag{13}$$

is used. Another possible limiting strategy is the subgradient aggregation strategy [20]. In Step C, also the parameter $u_{i,k+1}$ is updated and this can be done, for example with a weight updating algorithm presented in [13].

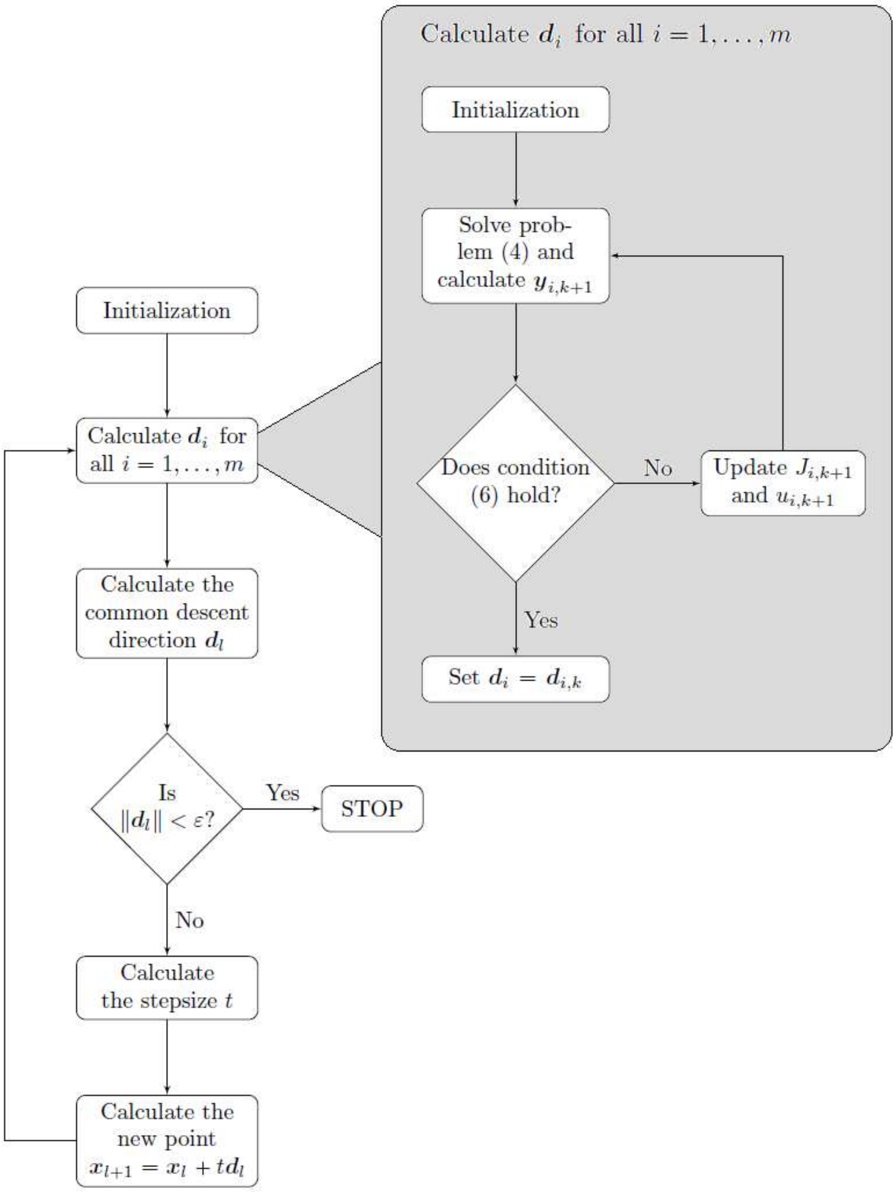

Figure 1: Flow chart of MSGDB

Next we prove that the solution generated by MSGDB is Pareto stationary. As mentioned before, in case of convex functions Pareto stationarity is equal to weak Pareto optimality.

Theorem 3.5. *Let us consider the problem* (1). *If MSGDB stops with a finite number of iterations, then the solution is Pareto stationary. On the other hand, any accumulation point of the infinite sequence of solutions generated by MSGDB is Pareto stationary.*

Proof. Assume that Algorithm 1 stops with a finite number of iterations and the stopping parameter ε is selected to be zero. Then $\|\boldsymbol{d}_l\| = 0$ and thus $\boldsymbol{d}_l = \boldsymbol{0}$. According to Theorem 3.4 the solution is Pareto stationary.

Suppose then that Algorithm 1 generates the infinite sequence of solutions $\{\boldsymbol{x}_l\}$ and $\boldsymbol{x}^*$ is the accumulation point of this sequence. Then there exists a convergent subsequence $\{\boldsymbol{x}_{l_s}\}$ with limit point $\boldsymbol{x}^*$. Therefore it is known that $\boldsymbol{\xi}^*_{i,l_s} \in \partial f_i(\boldsymbol{x}_{l_s})$ by Algorithm 1. Notate the accumulation point of the sequence $\{\boldsymbol{\xi}^*_{i,l_s}\}$ by $\boldsymbol{\xi}^*_i$. This accumulation point exists since the function f_i is locally Lipschitz continuous. Thus, there exists an index $\hat{s}$ such that for all indexes $s \geq \hat{s}$ the point $\boldsymbol{x}_{l_s} \in B(\boldsymbol{x}^*, \delta)$ and $|f_i(\boldsymbol{x}_{l_s}) - f_i(\boldsymbol{x}^*)| \leq K_i\|\boldsymbol{x}_{l_s} - \boldsymbol{x}^*\|$, where K_i is the Lipschitz constant of f_i. By Theorem 2.1.5 in [20] $\partial f_i(\boldsymbol{x}_{l_s}) \subset B(0, K_i)$ and thus the sequence $\boldsymbol{\xi}^*_{i,l_s}$ is bounded implying that the accumulation point exists. In addition, according to Theorem 2.1.5 in [20] we have $\boldsymbol{\xi}^*_i \in \partial f_i(\boldsymbol{x}^*)$.

Let $\boldsymbol{p}^*$ be a minimum norm element of the convex hull of subgradients i.e. $\boldsymbol{p}^* =$ argmin conv $\{\|\boldsymbol{\xi}^*_i\|\}$, $i = 1, \ldots, m$. The vector $\boldsymbol{p}^*$ is also the accumulation point of the sequence $\{\boldsymbol{p}_l\}$, where $\boldsymbol{p}_l$ is of form $\boldsymbol{p}_l =$ argmin conv $\{\|\boldsymbol{\xi}^*_{i,l_s}\|\}$, since $\boldsymbol{\xi}^*_i$ is an accumulation point of $\boldsymbol{\xi}^*_{i,l_s}$. Thus $\boldsymbol{d}^* = -\boldsymbol{p}^*$ is the common descent direction calculated at the point $\boldsymbol{x}^*$ according to Theorem 3.4.

If $\boldsymbol{d}^* = \boldsymbol{0}$, then the accumulation point $\boldsymbol{x}^*$ is Pareto stationary according to Theorem 3.4. Let now $\boldsymbol{d}^* \neq \boldsymbol{0}$ and assume that there exists sequence t_l with accumulation point t^*. Now there exists an index $\hat{l}$ such that for all $l > \hat{l}$ we have $\frac{1}{2}\|t^*\boldsymbol{d}^*\|$ being lower bound for $\|t_l\boldsymbol{d}_l\|$ and thus

$$\sum_{l=\hat{l}}^{\infty} \|t_l\boldsymbol{d}_l\| \geq \sum_{l=\hat{l}}^{\infty} \frac{1}{2}\|t^*\boldsymbol{d}^*\| = \infty.$$

Now we can conclude that $\boldsymbol{x}^*$ cannot be an accumulation point implying that the assumption $\boldsymbol{d}^* \neq \boldsymbol{0}$ does not hold. Thus the accumulation point of the infinite sequence of solutions is Pareto stationary.

□

3.2 Multiobjective proximal bundle method

Next we recall ideas of the multiobjective proximal bundle method [19, 22] which is a generalization of a single-objective proximal bundle method [13, 20]. It combines the ideas of the proximal bundle and the multiobjective linearization technique [27].

In MSGDB the problem (1) was approached by calculating descent directions for every objective function separately by utilizing the bundle idea and then by combining this information, a common descent direction was concluded. In MPB the bundle idea is also used but in a different way. A common descent search direction for all the objectives is formed straight with a different linearization technique. This linearization technique is based on [11, 27].

At first we introduce a concept of improvement function giving a tool to handle several objectives simultaneously. In unconstrained case the improvement function $H : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}$ is defined by

$$H(\boldsymbol{x}, \boldsymbol{y}) = \max_{i=1\ldots,m} \{f_i(\boldsymbol{x}) - f_i(\boldsymbol{y})\}\,. \tag{14}$$

According to [22] the problem (1) attains a weakly Pareto optimal solution at the point $\boldsymbol{x}^*$ if and only if

$$\boldsymbol{x}^* = \operatorname*{argmin}_{\boldsymbol{x}\in\mathbb{R}^n} H(\boldsymbol{x}, \boldsymbol{x}^*).$$

Thus at iteration k we are looking for a direction $\boldsymbol{d}_k$ which is a solution of the problem

$$\begin{aligned} \min \quad & H(\boldsymbol{x}_k + \boldsymbol{d}, \boldsymbol{x}_k) \\ \text{s.t.} \quad & \boldsymbol{d} \in \mathbb{R}^n. \end{aligned} \tag{15}$$

The problem (15) can be approximated by defining a convex piecewise linear approximation to improvement function (14). This approximation can be defined by

$$\hat{H}_k(\boldsymbol{x}) = \max_{i=1,\ldots,m} \left\{\hat{f}_i^k(\boldsymbol{x}) - f_i(\boldsymbol{x}_k)\right\},$$

where the function $\hat{f}_i$ is the same cutting plane model than in (2). Hence an approximation for the problem (15) is obtained and a search direction can be calculated by solving the problem

$$\boldsymbol{d}_k = \operatorname*{argmin}_{\boldsymbol{d}\in\mathbb{R}^n} \left\{\hat{H}_k(\boldsymbol{x}_k + \boldsymbol{d}) + \frac{1}{2} u_k \left\|\boldsymbol{d}\right\|^2\right\}, \tag{16}$$

where $u_k > 0$ is a weighting parameter as in (4). This nonsmooth problem can also be written as smooth quadratic problem like (5) for MSGDB and the following problem

$$\begin{aligned} \min \quad & v + \frac{1}{2} u_k \|\boldsymbol{d}\|^2 \\ \text{s.t.} \quad & -\alpha_{i,j}^k + \boldsymbol{\xi}_{i,j}^T \boldsymbol{d} \leq v, \qquad \text{for all } i = 1, \dots, m, \ \text{for all } j \in J_k, \end{aligned} \tag{17}$$

where the linearization error $\alpha_{i,j}^k$ is defined as in (3). The difference between problems (5) and (17) is that in the problem (5) there exist constraints only for the current value of i and in the problem (17) there exist constraints for every index i. Likewise the problem (5), the problem (17) can also be dualized to make it easier to solve.

In MSGDB, the stepsize was the largest positive number such that all the objective functions are decreasing. In MBP an another algorithm, called two-point line search strategy, is utilized to calculate stepsize. The aim of the two-point line search strategy is to find a stepsize $0 < t_k \leq 1$ such that $H(\boldsymbol{x}_k + t_k \boldsymbol{d}_k, \boldsymbol{x}_k)$ is minimal when $\boldsymbol{x}_k + t_k \boldsymbol{d}_k \in \mathbb{R}^n$. This stepsize is produced by the line search algorithm in [20] (pp. 126–130).

The general description of the algorithm of MPB is given below. Here the weight updating algorithm presented in [13] is used.

Algorithm 2. Multiobjective proximal bundle method (MPB)

Step 1: (*Initialization*) Select the starting point $\boldsymbol{x}_1$, the final accuracy tolerance $\varepsilon > 0$, the weight u_1 and line search parameters. Set an iteration index $k = 1$.

Step 2: (*Direction finding*) Calculate the search direction $\boldsymbol{d}_k$ from the problem (16).

Step 3: (*Stopping criterion*) Stop, if stopping criteria $-\frac{1}{2} v_k < \varepsilon$ is met.

Step 4: (*Line search*) Calculate a stepsize t_k by using two-point line search strategy and calculate the new point $\boldsymbol{x}_{k+1}$ and the trial point $\boldsymbol{y}_{k+1}$.

Step 5: (*Update*) Add more information to the bundle by evaluating $\boldsymbol{\xi}_{i,k+1} \in f_i(\boldsymbol{y}_{i,k+1})$ and adding a new index $k+1$ to the set J_k to improve the approximation. Update the weight parameter u_{k+1}. Go to step 2.

The solutions of MPB are weakly Pareto optimal as we see in the next theorem.

Theorem 3.6. [19] *Let us consider the problem* (1). *If MPB stops with a finite number of iterations, then the solution is weakly Pareto optimal. On the other hand, any accumulation point of the infinite sequence of solutions generated by MPB is weakly Pareto optimal.*

4 Computational experiments

In this section, we numerically compare the methods described in Section 3. At first we compare the search directions generated by the methods in order to notice that the search directions obtained with different methods are not necessarily the same direction. After that, we describe the implementations of the methods and give some computational examples and analyze the results.

4.1 Comparing search directions

At first we consider two simple examples where we calculate the search directions which we obtain at the first iteration round. One search direction is calculated with MSGDB and one with MPB. We apply two different types of weighting parameters u_k, one with $u_k = 2u_{k-1}$, where $u_1 = 1$ and the other with $u_k = 1$ for all k. After that, we calculate stepsizes. In these examples in Section 4.1 we use the exact line search.

Both example problems are of form

$$\begin{aligned} \min \quad & \{f_1(\boldsymbol{x}), f_2(\boldsymbol{x})\} \\ \text{s.t.} \quad & \boldsymbol{x} \in \mathbb{R}^2. \end{aligned} \tag{18}$$

In the first problem the convex objective functions in the problem (18) are

$$\begin{aligned} f_1(\boldsymbol{x}) &= \max\left\{x_1^2 + (x_2-1)^2,\ (x_1+1)^2\right\} \\ f_2(\boldsymbol{x}) &= \max\left\{2x_1 + 2x_2,\ x_1^4 + x_2^2\right\} \end{aligned}$$

and the function values at the starting point $\boldsymbol{x}_1 = (0,2)^T$ are $f_1(\boldsymbol{x}_1) = 1$ and $f_2(\boldsymbol{x}_1) = 4$. We get the results shown in Table 1.

Table 1: The first example

	$u_k = 2u_{k-1}$		$u_k = 1$	
	MSGDB	MPB	MSGDB	MPB
$\boldsymbol{d}$	$(0.5000, 0.5000)$	$(0.3824, 0.5294)$	$(0.4000, 0.8000)$	$(0.4000, 0.8000)$
t	1.0000	1.0064	1.1335	1.1335
$\boldsymbol{x}_2$	$(-0.5000, 0.5000)$	$(-0.3848, 0.5294)$	$(-0.4534, 1.0932)$	$(-0.4534, 1.0932)$
$f_1(\boldsymbol{x}_2)$	0.5000	0.3785	0.2988	0.2988
$f_2(\boldsymbol{x}_2)$	2.3125	0.2923	1.2796	1.2796

In the second problem the convex objective functions in the problem (18) are

$$\begin{aligned} f_1(\boldsymbol{x}) &= \max\left\{(x_1-2)^2 + (x_2+2)^2, x_1^2 + 8x_2\right\} \\ f_2(\boldsymbol{x}) &= \max\left\{5x_1 + x_2, x_1^2 + x_2^2\right\} \end{aligned}$$

Table 2: The second example

	$u_k = 2u_{k-1}$		$u_k = 1$	
	MSGDB	MPB	MSGDB	MPB
$\boldsymbol{d}$	$(2.0558, 3.0107)$	$(0.4122, 0.1765)$	$(1.9527, 2.9901)$	$(1.8459, 1.5986)$
t	0.6077	1.4612	0.6220	0.7628
$\boldsymbol{x}_2$	$(-0.2493, 0.1704)$	$(0.3977, 1.7421)$	$(-0.2146, 0.1402)$	$(-0.4081, 0.7806)$
$f_1(\boldsymbol{x}_2)$	9.7700	16.5700	9.4849	13.5307
$f_2(\boldsymbol{x}_2)$	0.2326	3.7306	0.0657	0.7759

and the function values at the starting point $\boldsymbol{x}_1 = (1, 2)^T$ are $f_1(\boldsymbol{x}_1) = 17$ and $f_2(\boldsymbol{x}_2) = 7$. We get the results shown in Table 2.

From these two examples we can notice that with MSGDB and MPB we do not necessarily obtain the same search directions. For example in Table 1 we have two cases with different weighting parameters. In the first case directions are different and in the second case we obtain the same directions.

In addition we cannot say which one is better way to calculate directions. As we see, in the first case of the first example the direction calculated with MPB gives a better point than the direction calculated with MSGDB. In this case, the better point refers to the point where both objective functions f_1 and f_2 obtain smaller value. However in both cases of the second example MSGDB gives a better point than MPB.

Thus based on the way to calculate the search direction we cannot say that one method is always better than another.

4.2 Implementation and numerical results

In numerical experiments, we have used single-objective convex test problems CB3, DEM, QL, LQ, Mifflin1 and Wolfe described in [16] and combined these functions in order to obtain twenty multiobjective problems. The used combinations are described in Table 3. All our test problems are nonsmooth and convex. The dimension of all test problems is two. In the first 15 problems we have two objectives and the last five problems have three objectives.

We have used the implementation of MPB described in [18], where two-point line search algorithm is employed. In MSGDB we apply Armijo type rule [1] as the line search due to its simplicity. Both the methods are implemented in Fortran. To make the methods more comparable we have used the same quadratic solver described in [15] with both methods. In order to update the weighting parameter u_k the weight updating algorithm described in [13] is used. In both methods the size of the bundle is bounded by using the set (13) and the value of J_{max} is chosen to be $n + 3$. In following we consider one test problem closer and after that we analyze the results of several tests.

Table 3: Test problems

No.	Problems	No.	Problems
1.	CB3 & DEM	11.	QL & Mifflin1
2.	CB3 & QL	12.	QL & Wolfe
3.	CB3 & LQ	13.	LQ & Mifflin1
4.	CB3 & Mifflin1	14.	LQ & Wolfe
5.	CB3 & Wolfe	15.	Mifflin1 & Wolfe
6.	DEM & QL	16.	CB3, DEM & QL
7.	DEM & LQ	17.	LQ, Mifflin1 & Wolfe
8.	DEM & Mifflin1	18.	DEM, QL & LQ
9.	DEM & Wolfe	19.	CB3, Mifflin1 & Wolfe
10.	QL & LQ	20.	DEM, LQ & Wolfe

Let us take a closer look at the test problem number 3. In that problem objective functions are combined from test problems CB3 and LQ [16]. Thus objective functions of the problem (18) are now

$$f_1(\boldsymbol{x}) = \max\{x_1^4 + x_2^2, (2 - x_1)^2 + (2 - x_2)^2, 2e^{x_2 - x_1}\}$$
$$f_2(\boldsymbol{x}) = \max\{-x_1 - x_2, -x_1 - x_2 + x_1^2 + x_2^2 - 1\}.$$

The starting point is chosen to be $\boldsymbol{x}_1 = (2, 2)^T$, the line search parameter $m_L = 0.25$ and the stopping parameter $\varepsilon = 10^{-5}$. The performance of MSGDB is described in Table 4, where current points and the function values at those points are listed at every iteration. As we see, the value of both objective functions decreases at every iteration.

Table 4: The performance of the MSGDB algorithm with test problem 3

Iteration	$\boldsymbol{x}$	$f(\boldsymbol{x})$
1	(2, 2)	(20, 3)
2	(1.07040, 2.19346)	(6.14850, 1.69316)
3	(1.01611, 1.22948)	(2.57763, -0.70149)
4	(0.94961, 1.04304)	(2.19586, -1.00295)
5	(0.95189, 0.98783)	(2.12303, -1.05782)

The performance of MSGDB in this test problem is also illustrated in Figure 2. In this figure, gray contours correspond the contours of the first objective function while dashed gray contours correspond the contours of the second one. The optimal points of the first and second objective are marked with black and white square, respectively. The value of the function at the optimal point of the first objective is $f = (2, -1)$ and at the optimal point of the second objective $f = (3.34, -1.41)$. The black point represents the solution $\boldsymbol{x}_5$ obtained with

MSGDB and circles are previous iteration points. Now we can see that the solution obtained is closer to the optimal point of the first objective function than the optimal point of the second objective function.

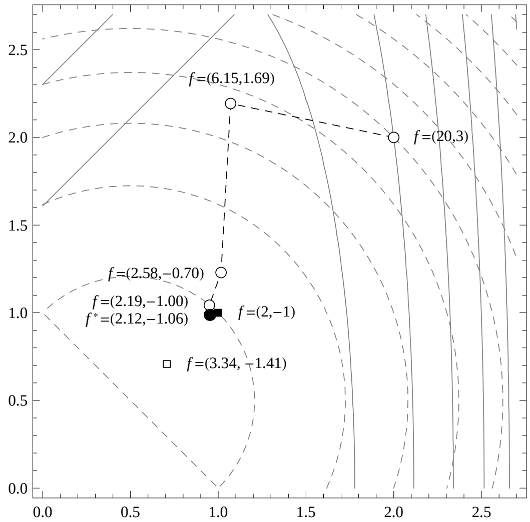

Figure 2: The performance of the algorithm in decision space for testproblem 3

In Figures 3 and 4, the situation, where the algorithm is run ten times with different starting points is illustrated. In Figure 3, we have solutions obtained in the decision space marked with black points. Also four paths of the algorithm are illustrated with black dashed lines in order to demonstrate the performance of the algorithm. In Figure 4, these solutions are depicted in the objective space. Again, in Figures 3 and 4, squares represent single-objective optimal points.

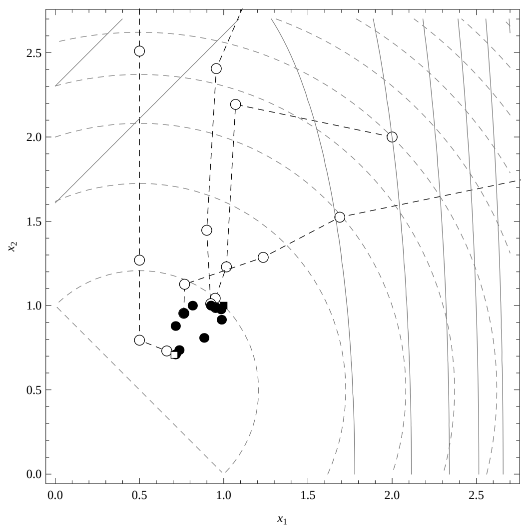

Figure 3: The solutions obtained in decision space for test problem 3 with several starting points

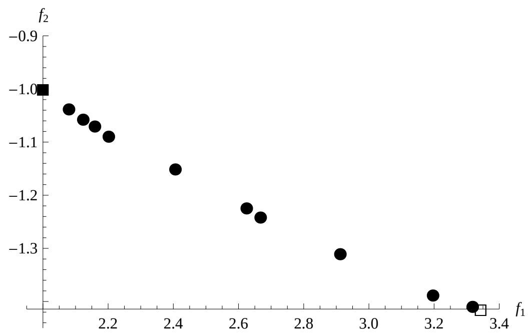

Figure 4: The solutions obtained in objective space for test problem 3 with several starting points

From Figures 3 and 4 we can observe that the solution obtained depends on the starting point. With different starting points we can generate different (weakly) Pareto optimal solutions and obtain an approximation of the Pareto optimal set.

In Table 5 the optimal function values for all the testproblems obtained with MSGDB and MPB are described. The same starting points are used and the stopping criteria are set such that $\varepsilon = 10^{-5}$ in both methods. In addition, the number of iterations and function calls are listed. In the implementation of MPB objective functions are called at the same time and in the implementation of MSGDB all objective functions are called separately. Thus there are three function call columns (F1, F2 and F3) for MSGDB and only one column (F) for MPB in Table 5.

From results in Table 5 we can conclude that the number of iterations are approximately the same order since according to the results the average iterations needed for MSGDB is 9.00 and for MPB is 11.20. Even the number of iterations is slightly smaller with MSGDB, the number of function calls in this implementation is larger than function calls needed with MPB but they are still same magnitude. We can also observe that the methods produce mainly different weakly Pareto optimal solutions since the average relative distance of solutions in the objective space is 0.56 varying in the interval from 0.00 to 2.62 and only once they obtain the same weakly Pareto optimal solution. In addition, it is worth noting that the implementation of MPB is the result of long development and testing process contrary to the implementation of MSGDB being only the first implementation.

Table 5: Results of numerical tests

	MSGDB					MPB		
No.	F1	F2	F3	Iter.	$f(\boldsymbol{x}^*)$	F	Iter.	$f(\boldsymbol{x}^*)$
1.	24	13		3	$(3.147, 4.432)$	8	7	$(4.106, 3.169)$
2.	43	29		4	$(7.834, 7.200)$	12	11	$(6.495, 10.217)$
3.	22	17		5	$(2.123, -1.058)$	5	4	$(2.030, -1.015)$
4.	52	37		11	$(2.135, 16.475)$	19	18	$(2.065, 17.736)$
5.	87	64		21	$(2.000, 24.995)$	19	18	$(4.274, 13.057)$
6.	35	26		4	$(16.800, 7.200)$	8	7	$(16.800, 7.200)$
7.	14	10		2	$(2.814, -0.938)$	7	6	$(2.958, -1.068)$
8.	12	12		3	$(3.500, -0.750)$	8	6	$(-1.251, 11.280)$
9.	36	95		9	$(1.519, 11.129)$	12	11	$(2.645, -4.757)$
10.	27	27		4	$(7.200, 2.600)$	7	6	$(7.424, 2.507)$
11.	27	39		4	$(7.200, 122.800)$	15	14	$(7.256, 122.439)$
12.	78	58		19	$(7.200, 49.200)$	13	12	$(8.582, 47.529)$
13.	41	107		10	$(-1.386, -0.553)$	32	31	$(-1.373, -0.856)$
14.	86	67		22	$(-1.414, 17.678)$	18	17	$(-1.039, 12.471)$
15.	49	46		9	$(4.064, -7.777)$	21	13	$(-0.978, 14.681)$
16.	18	12	16	3	$(7.827, 16.793, 7.214)$	5	4	$(6.331, 14.886, 10.977)$
17.	93	171	74	24	$(-1.414, -0.707, 17.678)$	19	18	$(-1.039, -0.665, 12.471)$
18.	9	11	10	2	$(13.958, 12.925, 1.379)$	7	6	$(16.307, 7.618, 2.461)$
19.	82	64	59	19	$(2.000, 19.000, 25.000)$	13	12	$(2.856, 6.212, 19.623)$
20.	22	9	16	2	$(0.325, 3.3697, 21.199)$	4	3	$(3.000, -1.000, 12.500)$
Av:	42.85	45.70	35.00	9.00		12.60	11.20	

5 Conclusions

We have proposed a new descent bundle based method for convex unconstrained multiobjective optimization. The method generalizes ideas from the multiple-gradient descent algorithm and combines them with the proximal bundle method. In order to find a common descent direction for all the objectives the idea of the multiple-gradient descent algorithm is utilized and in order to obtain descent directions for each objective separately the idea of the proximal bundle is used. Thus in the case of a single-objective function, the search direction generated with MSGDB is similar to the search direction generated with the proximal bundle method and in the case of differentiable objective functions MSGDB is similar to the multiple-gradient descent algorithm.

We have described the basic idea of MPB as a reference method. MPB is chosen for reference method since it is also a descent method for multiobjective optimization utilizing the bundle idea. When in MSGDB the bundle idea is used in order to find the descent direction for all objectives separately and after that to find one common descent direction, in MPB all objectives are taken into consideration at the same time with the improvement function and the bundle idea is used in order to find a descent direction for this improvement function.

We have seen that the methods described may produce different directions and we cannot say that one would always be better than another. According to numerical experiments, we have shown that the number of iterations needed with MSGDB is small and the same order that is needed with MPB. However, the number of function calls needs to be improved in the implementation of MSGDB. This could be done with better line search algorithm.

In addition, we observed that the methods usually produce different weakly Pareto optimal solutions. In interactive methods, it usually is useful to have several different solutions produced from the same starting point. Thus these kind of descent methods are needed, for example, in interactive methods.

In order to extend MSGDB in future the aim is to design a method which is able to solve also nonconvex and constrained multiobjective problems. Another possible development could be the invocation of the separately calculated search directions. Since every objective function has own search direction, those might be used, for instance, in interactive methods by scaling directions according to the decision maker's preferences.

Acknowledgements. The research has been financially supported by the Finnish Academy of Science and Letters (the Vilho, Yrjö and Kalle Väisälä Foundation).

References

[1] L. Armijo. Minimization of Functions Having Lipschitz Continuous First Partial Derivatives. *Pacific Journal of Mathematics*, 16(1):1–3, 1966.

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

[3] M. S. Bazaraa and C. M. Shetty. *Nonlinear Programming: Theory and Algorithms*. John Wiley & Sons, Inc., New York/Jersey, 1979.

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

[5] J.-A. Désidéri. Multiple-Gradient Descent Algorithm (MGDA). Technical Report 6953, INRIA Research Report, 2009.

[6] J.-A. Désidéri. Multiple-Gradient Descent Algorithm (MGDA) for Multiobjective Optimization. *Compte rendus de l'Académie des sciences, Ser. I*, 350:313–318, 2012.

[7] M. Ehrgott. *Multicriteria Optimization*. Springer, Berlin, second edition, 2005.

[8] J. Fliege and B. F. Svaiter. Steepest Descent Methods for Multicriteria Optimization. *Mathematical Methods of Operations Research*, 51(3):479–494, 2000.

[9] K. Harada, J. Sakuma, and S. Kobayashi. Local Search for Multiobjective Function Optimization: Pareto Descent Method. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2006)*, pages 659–666, 2006.

[10] J. Haslinger and P. Neittaanmäki. *Finite Element Approximation for Optimal Shape, Material and Topology Design*. J. Wiley & Sons, Chichester, 1996.

[11] K. C. Kiwiel. A Descent Method for Nonsmooth Convex Multiobjective Minimization. *Large Scale Systems*, 8(2):119–129, 1985.

[12] K. C. Kiwiel. *Methods of Descent for Nondifferentiable Optimization*. Springer-Verlag, Berlin, 1985.

[13] K. C. Kiwiel. Proximity Control in Bundle Methods for Convex Nondifferentiable Optimization. *Mathematical Programming*, 46:105–122, 1990.

[14] C. Lemaréchal. Nondifferentiable Optimization. In G. G. Nemhauser, A. H. G. Rinnooy Kan, and M. J. Todd, editors, *Optimization*, pages 529–572. Elsevier North-Holland, Amsterdam, 1989.

[15] L. Lukšan. Dual Method for Solving a Special Problem of Quadratic Programming as a Subproblem at Linearly Constrained Nonlinear Minimax Approximation. *Kybernetika*, 20:445–457, 1984.

[16] L. Lukšan and J. Vlcek. Test problems for nonsmooth unconstrained and linearly constrained optimization. Technical Report 798, Institute of Computer Science, Academy of Sciences of the Czech Republic, Prague, 2000.

[17] M. M. Mäkelä. *Survey Of Bundle Methods for Nonsmooth Optimization*. *Optimization Methods and Software*, 17(1):1–29, 2002.

[18] 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.

[19] M. M. Mäkelä, N. Karmitsa, and O. Wilppu. Multiobjective Proximal Bundle Method for Nonsmooth Optimization. Technical Report 1120, TUCS Technical Reports, Turku Centre for Computer Science, Turku, 2014.

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

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

[22] K. Miettinen and M. M. Mäkelä. Interactive Bundle-based Method for Nondifferentiable Multiobjective Optimization: NIMBUS. *Optimization*, 34:231–246, 1995.

[23] 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.

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

[25] H. Mukai. Algorithms for Multicriterion Optimization. *IEEE Transactions on Automatic Control AC-25*, pages 177–186, 1979.

[26] 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.

[27] 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

Turku School of Economics
- Institute of Information Systems Sciences

Åbo Akademi University
- Department of Computer Science
- Institute for Advanced Management Systems Research

ISBN 978-952-12-3154-4
ISSN 1239-1891