Preprint No. M **18/03**

A branch-and-bound based algorithm for nonconvex multiobjective optimization

Julia Niebling, Gabriele Eichfelder

Februar 2018

URN: urn:nbn:de:gbv:ilm1-2018200024

A Branch-and-Bound based Algorithm for Nonconvex Multiobjective Optimization

Julia Niebling[*], Gabriele Eichfelder[††]

February 19, 2018

Abstract

A new branch-and-bound based algorithm for smooth nonconvex multiobjective optimization problems with convex constraints is presented. The algorithm computes an (ε, δ)-approximation of all globally optimal solutions. We introduce the algorithm which uses selection rules, discarding and termination tests. The discarding tests are the most important aspect, as they examine in different ways whether a box can contain optimal solutions and determine by that the speed and effectiveness of the algorithm. We present a discarding test which combines techniques from the αBB method from global scalar-valued optimization with outer approximation techniques from convex multiobjective optimization and the concept of local upper bounds from combinatorial multiobjective optimization. We apply the algorithm to several test instances as well as to an application in Lorentz force velocimetry.

Key Words: Multiobjective Optimization, Nonconvex Optimization, Global Optimization, Branch-and-Bound Algorithm, αBB-Method

Mathematics subject classifications (MSC 2010): 90C26, 90C29, 90C30

1 Introduction

For multiobjective optimization problems (MOPs) meanwhile a variety of algorithms exist, [5, 13, 15, 32]. Such problems appear in engineering or economics each time when various objective functions have to be minimized simultaneously. In general there is no point which minimizes all objective functions at the same time and thus one uses another optimality concept than the one in scalar optimization. However, just like in scalar optimization, we have to distinguish between local and global optimal solutions. While local solutions are only optimal in a neighborhood, global solutions have to be optimal on the whole feasible set. Most algorithms for multiobjective optimization problems aim on finding local solutions only and are thus only appropriate for convex problems.

[*]Institute for Mathematics, Technische Universität Ilmenau, Po 10 05 65, D-98684 Ilmenau, Germany, `julia.niebling@tu-ilmenau.de`.

[††]Institute for Mathematics, Technische Universität Ilmenau, Po 10 05 65, D-98684 Ilmenau, Germany, `gabriele.eichfelder@tu-ilmenau.de`.

Algorithms for solving (convex) multiobjective optimization problems are highly investigated, see for an overview for example [11, 22]. Most of these algorithms are based on scalarization approaches, [13, 15, 17, 34]. In these techniques a new scalar-valued problem depending on some parameters is formulated and solved by known methods for scalar-valued optimization problems. With other parameters and more iterations an approximation of the optimal solution set can be obtained. If we use only local methods to find optimal solutions of the nonconvex scalar-valued problems we will also get only local solutions of the original vector-valued problem. If we use a global solver for each choice of parameters for the scaralized problems then this is a very time consuming and inefficient approach. Moreover, most scalarizations turn some of the nonconvex objective functions into constraints, but it is well known that nonconvex constraints are especially difficult to handle [24]. For instance the weighted sum methods avoids this, but it is also well known that this is not an appropriate method for nonconvex problems. For that reason the development of direct global solvers for MOPs without using scalarizations is important. For an introduction into global optimization see for example [20].

Many methods for global optimization use stochastic strategies. Common used algorithms are evolutionary algorithms which are trying to find global minima by crossing and mutating of individuals of a constructed population, i.e. some feasible points of the optimization problem [8, 9, 16, 42]. It is known that these procedures are able to find a global minimum in an infinite amount of time, but they do not guarantee to find a good solution in a finite time. That is the reason why it is of special interest to propose also a deterministic algorithm.

An approach to find global solutions for MOPs deterministically was introduced by Jahn, [21]. There the whole search space is discretized and refined into 'promising' areas. But in higher dimensions of the pre-image space, i.e. $n \geq 3$, a large number of function evaluations is required. This is due to the fact that Jahn's method does not use any information about derivatives. Another derivative-free algorithm for MOPs was proposed by Custódio and Madeira in [7] which is based on a direct search approach with a clever multistart strategy and is also able to find global solutions.

Other global optimization algorithms are based on branch-and-bound methods, for example [1, 2, 6, 10, 19, 33, 41, 43]. For scalar-valued optimization problems two of the most well-known algorithms, which use box partitions, are the DIRECT algorithm [23] and the αBB-method [33]. While the DIRECT algorithm focuses on selecting boxes to have a good balance between local and global search but can also not guarantee to find good solutions in finite time [29], the αBB-algorithm uses lower bounds of the global minimum obtained by minimizing a convex underestimator and improves them until a given accuracy is reached. Other methods to find lower bounds for global minima use maximal values of the dual problem, [10], or by using the Lipschitz constant, [28, 44]. The DIRECT algorithm was also extended to MOPs, see for example [6, 43]. However, in most cases the plain multiobjective version of the DIRECT algorithm shows bad convergence results and has to be accelerated by another global or local optimization method. The first branch-and-bound based algorithm for more than one objective function and with first convergence results was introduced by Fernández and Tóth in [14]. The described procedure is for biobjective optimization problems. Recently, see [44] and also [36], another algorithm for biobjective problems was proposed by Žilinskas and Žilinskas. They use the Lipschitz property of the objective functions and iterative trisections of the search domain which is assumed to be a box. A possible extension to more general constraints is at least not obvious.

We base our algorithm on a branch-and-bound approach as the one proposed in [14]. However, we give an algorithm for an arbitrary number of objective functions and we provide new discarding tests based on the concept of convex underestimators from the αBB method [33]. This results in convex MOPs which have to be considered on each subbox. We give improved lower bounds for them by using approaches from convex multiobjective optimization combined with a versatile concept from multiobjective optimization. Finally, we are able to prove that we find an approximation of the set of globally optimal solutions for the MOPs with predefined quality in finite time. This paper starts with the basics of multiobjective and global optimization in Section 2. In Section 3, the new algorithm will be introduced after presenting a typical Branch-and-Bound algorithm for MOPs. In Section 4 we prove the correctness and termination of the proposed algorithm. Since we propose the algorithm for boxconstrained problems we will describe the handling of constraints in Section 5. Numerical results, also for an application problem from Lorentz force velocimetry, are presented in Section 6. We end with a conclusion and an outlook in Section 7.

2 Basics of Multiobjective and Global Optimization

In this section we introduce the basic definitions and concepts which we need for the new algorithm. We study the following MOP:

$$\min_{x\in X^0} f(x) = (f_1(x),\dots,f_m(x))^T \text{ s.t. } g_r(x) \le 0 \text{ for all } r = 1,\dots,p \qquad (MOP)$$

where $f_j\colon \mathbb{R}^n \to \mathbb{R},\ j = 1,\dots,m$ are twice continuously differentiable functions and $g_r\colon \mathbb{R}^n \to \mathbb{R},\ r = 1,\dots,p$ are continuously differentiable convex functions. We do not discuss the case of nonconvex constraints here, as the same difficulties [24] and strategies as in the scalar-valued case arise and we want to concentrate on the multiobjective aspects within this paper. We assume that at least one of the objective functions is nonconvex. The set $X^0 \subseteq \mathbb{R}^n$ is assumed to be a nonempty *box* (also called hyper rectangle), i.e. $X^0 = \{x \in \mathbb{R}^n \mid \underline{x} \le x \le \overline{x}\}$ with two points $\underline{x}, \overline{x} \in \mathbb{R}^n$. Thereby we write $x \le y$ if $x_i \le y_i$ for all $i = 1,\dots,n$ and $x < y$ if $x_i < y_i$ for all $i = 1,\dots,n$. Moreover, the feasible set $M := \{x \in X^0 \mid g_r(x) \le 0,\ r = 1,\dots,p\}$ is assumed to be nonempty. For an arbitrary set $A \subseteq M$ we define the image set of A under f by $f(A) := \{f(x) \in \mathbb{R}^m \mid x \in A\}$.

The common optimality concept in multiobjective optimization is *efficiency*: A point $x^* \in M$ is said to be *efficient* for (MOP) if there does not exist another $x \in M$ such that $f(x) \le f(x^*)$ and $f(x) \ne f(x^*)$. The set of all efficient points is called *efficient set* and is denoted by X_E. We say x^1 *dominates* x^2 if $x^1, x^2 \in M$, $f(x^1) \le f(x^2)$ and $f(x^1) \ne f(x^2)$. We can define similar terms in the image space. Let $x^* \in M$. A point $y^* = f(x^*)$ is said to be *nondominated* for (MOP) if x^* is efficient for (MOP). The set of all nondominated points is called *nondominated set*. We say y^1 *dominates* y^2 if $y^1, y^2 \in \mathbb{R}^m$, $y^1 \le y^2$ and $y^1 \ne y^2$. Moreover, we say y^1 *strictly dominates* y^2 if $y^1, y^2 \in \mathbb{R}^m$ and $y^1 < y^2$.

The aim of the new algorithm is to find an (ε, δ)-minimal set $\mathcal{A}$ of (MOP), which is defined next. Let e denote the m-dimensional all-one vector $(1, 1, \dots, 1)^T \in \mathbb{R}^m$ and $\|\cdot\|$ denotes the Euclidean norm.

Definition 2.1. *Let $\varepsilon \ge 0$ and $\delta \ge 0$ be given.*

(i) A point $\bar{x} \in M$ is an ε-minimal point of (MOP), if there does not exist another $x \in M$ with $f(x) \leq f(\bar{x}) - \varepsilon e$ and $f(x) \neq f(\bar{x}) - \varepsilon e$.

(ii) A set $\mathcal{A} \subseteq M$ is an (ε, δ)-minimal set of (MOP), if every point of $\mathcal{A}$ is an ε-minimal point of (MOP) and if for all $y \in X_E$ there is an $\bar{x} \in \mathcal{A}$ with $\|\bar{x} - y\| \leq \delta$.

This definition of ε-optimality was introduced in an equivalent version in [27]. A more general concept for approximate solutions for vector optimization can be found in [18]. Furthermore a slighty different definition of ε-optimality is presented in [31], where different accuracies for the objective functions are allowed.

We use some ideas and concepts of interval arithmetic in our algorithm. For an introduction to interval analysis we refer to [35]. The set of all n-dimensional real boxes will be denoted by $\mathbb{IR}^n$. The width of a box $X \in \mathbb{IR}^n$ is defined as $\omega(X) := \|\overline{x} - \underline{x}\|$. Interval arithmetic allows to calculate lower bounds of function values on a given box, see [14, 35]. Another method to calculate lower bounds was proposed for scalar-valued functions in [33] under the name αBB and is based on the concept of convex underestimators. A *convex underestimator* for a function $h\colon \mathbb{R}^n \to \mathbb{R}$ on a box $X = [\underline{x}, \overline{x}] \in \mathbb{IR}^n$ is a convex function $\widehat{h}\colon X \to \mathbb{R}$ with $\widehat{h}(x) \leq h(x)$ for all $x \in X$. A convex underestimator for a twice continuously differentiable function h on X can be calculated by $\widehat{h}(x) := h_\alpha(x) := h(x) + \frac{\alpha}{2}(\underline{x} - x)^T(\overline{x} - x)$, where $\alpha \geq \max\{0, -\min_{x \in X} \lambda_{\min}(x)\}$. Here, $\lambda_{\min}(x)$ denotes the smallest eigenvalue of the Hessian $H_h(x)$ of h in x, [33]. The minimum value of h_α over X, which can be calculated by standard techniques from convex optimization, delivers then a lower bound for the values of h on X. Clearly, if $\tilde{X} \subseteq X$ and h_α is a convex underestimator of h on X, then h_α is also a convex underestimator of h on $\tilde{X}$. With our numerical method to calculate h_α on X and $h_{\tilde{\alpha}}$ on $\tilde{X} \subseteq X$ we always obtain $\tilde{\alpha} \leq \alpha$. For simplicity of the presentation in this work we use only the parameter α which was calculated for the convex underestimator on the box $X = X^0$. Furthermore if we use the boundaries $\underline{\tilde{x}}$ and $\overline{\tilde{x}}$ of $\tilde{X} = [\underline{\tilde{x}}, \overline{\tilde{x}}] \subseteq X$ to define another convex underestimator $\tilde{h}_\alpha\colon \mathbb{R}^n \to \mathbb{R}$ on $\tilde{X}$ by $\tilde{h}_\alpha(x) := h(x) + \frac{\alpha}{2}(\underline{\tilde{x}} - x)^T(\overline{\tilde{x}} - x)$, we obtain immediately $\tilde{h}_\alpha(x) \geq h_\alpha(x)$ for all $x \in \tilde{X}$. A lower bound for $\lambda_{\min}(x)$ over X can be calculated easily with the help of interval arithmetic, see [33]. For that the Matlab toolbox Intlab can efficiently be used [37]. See also [40] for improved lower bounds. The above proposed and some other methods to obtain lower bounds for $\lambda_{\min}(x)$ are described and tested in [1]. There are also other possibilities for the calculation of convex underestimators. For example in [1] special convex underestimators for bilinear, trilinear, fractional, fractional trilinear or univariate concave functions were defined. Here, we restrict ourselves to the above proposed convex underestimator. The theoretical results remain true in case the above underestimators are replaced by tighter ones.

Remark 2.2. *[33] For all $\alpha \geq 0$ the maximal pointwise difference between h and h_α is $\frac{\alpha}{2}\omega(X)^2$, i.e. $\max_{x \in X} |h(x) - h_\alpha(x)| = \frac{\alpha}{2}\omega(X)^2$.*

In the next lemma we show that the distance between the minimal value of a convex underestimator and the other function values of a smooth function h over a box is bounded by a given $\varepsilon > 0$ if the box width is small enough.

Lemma 2.3. *Let a box $X \in \mathbb{IR}^n$, a twice continuously differentiable nonconvex function $h\colon \mathbb{R}^n \to \mathbb{R}$, a convex underestimator h_α of h on X and a positive scalar $\varepsilon > 0$ be given.*

Moreover let $L > 0$ be chosen such that $L \geq \sqrt{n}\left|\frac{\partial}{\partial x_i}h(x)\right|$ for all $i \in \{1,\dots,n\}$ and $x \in X$. Let $\tilde{X} = [\underline{\tilde{x}}, \overline{\tilde{x}}]$ be a box with $\tilde{X} \subseteq X$ and with

$$\omega(\tilde{X}) \leq -\frac{L}{\alpha} + \sqrt{\frac{L^2}{\alpha^2} + \frac{\varepsilon}{\alpha}} =: \delta_X \tag{1}$$

and define $\tilde{h}_\alpha \colon \mathbb{R}^n \to \mathbb{R}$ by $\tilde{h}_\alpha(x) := h(x) + \frac{\alpha}{2}(\underline{\tilde{x}} - x)^T(\overline{\tilde{x}} - x)$ which is a convex underestimator of h on $\tilde{X}$. Then for $v := \min\limits_{x \in \tilde{X}} \tilde{h}_\alpha(x)$ it holds $|h(x) - v| \leq \frac{\varepsilon}{2}$ for all $x \in \tilde{X}$.

Proof. Note that $\alpha \neq 0$ because of h being nonconvex. Let $\tilde{x}$ be a minimal solution of $\min_{x \in \tilde{X}} \tilde{h}_\alpha(x)$, i.e. $v = \tilde{h}_\alpha(\tilde{x})$. With Remark 2.2 it follows $|h(\tilde{x}) - v| = |h(\tilde{x}) - \tilde{h}_\alpha(\tilde{x})| \leq \frac{\alpha}{2}\omega(\tilde{X})^2$. Now let $x, y \in \tilde{X}$ be arbitrarily chosen. By the mean value theorem there exists $\xi \in \{\lambda x + (1-\lambda)y \in \mathbb{R}^n \mid \lambda \in (0,1)\}$ with $h(x) - h(y) = \nabla h(\xi)^T(x - y)$. Together with the Cauchy-Schwarz inequality we obtain $|h(x) - h(y)| \leq \|\nabla h(\xi)\| \|x - y\|$. As $\|\nabla h(\xi)\| = \sqrt{\sum_{i=1}^n \left(\frac{\partial}{\partial x_i}h(\xi)\right)^2} \leq \sqrt{\sum_{i=1}^n \frac{L^2}{n}} = L$ we get $|h(x) - h(y)| \leq L\omega(\tilde{X})$ for all $x, y \in \tilde{X}$. Now let $x \in \tilde{X}$ be arbitrarily chosen. Then it follows for the distance between v and $h(x)$ due to (1): $|h(x) - v| \leq |h(x) - h(\tilde{x})| + |h(\tilde{x}) - v| \leq L\omega(\tilde{X}) + \frac{\alpha}{2}\omega(\tilde{X})^2 \leq \frac{\varepsilon}{2}$. □

The constant L in the above lemma can be obtained by using techniques from interval arithmetic for instance with the help of the Matlab toolbox Intlab [37].

As we are considering vector-valued functions in this work, we use convex underestimators for every component function on its own. Thus we denote the vector-valued convex underestimator for the function $f \colon \mathbb{R}^n \to \mathbb{R}^m$ with $f_\alpha \colon \mathbb{R}^n \to \mathbb{R}^m$, $x \mapsto (f_{1,\alpha}(x), \dots, f_{m,\alpha}(x))^T$, where $f_{j,\alpha}$ is a convex underestimator of f_j, $j = 1,\dots,m$. Note that we choose for the convex underestimators for each objective function the same parameter α for simplicity of the presentation although different ones are of course also possible. Lemma 2.3 can be easily generalized for the vector-valued case by considering each objective function on its own.

3 The new Branch-And-Bound Based Algorithm

We introduce the algorithm for a clearer presentation first for box-constrained MOPs. The handling of constraints will be discussed in Section 5. Thus, in the following, we assume the MOP to be given as

$$\min_{x \in \mathbb{R}^n} f(x) = (f_1(x), \dots, f_m(x))^T \text{ s.t. } x \in X^0, \tag{P}$$

i.e. $M = X^0$. Algorithm 1 gives a basic branch-and-bound algorithm as it was already proposed in [14]:

Algorithm 1

INPUT: $X^0 \in \mathbb{R}^n, f \in \mathcal{C}^2(\mathbb{R}^n, \mathbb{R}^m)$
OUTPUT: $\mathcal{L}_S$

```
1: $\mathcal{L}_W \leftarrow \{X^0\}, \mathcal{L}_S \leftarrow \emptyset$
2: while $\mathcal{L}_W \neq \emptyset$ do
3:     Select a box $X^*$ from $\mathcal{L}_W$ and delete it from $\mathcal{L}_W$          Selection rule
4:     Bisect $X^*$ perpendicularly to a direction of maximum width $\rightarrow X^1, X^2$
5:     for $l = 1, 2$ do
6:         if $X^l$ cannot be discarded then                                      Discarding tests
7:             if $X^l$ satisfies a termination rule then                         Termination rule
8:                 Store $X^l$ in $\mathcal{L}_S$
9:             else Store $X^l$ in $\mathcal{L}_W$
```

The lists $\mathcal{L}_W$ and $\mathcal{L}_S$ are the working list and the solution list respectively. A typical selection rule is the one proposed in [14]:

Selection rule: *Select the box $X^* \in \mathcal{L}_W$ with a minimum lower bound of f_1.*

In our algorithm this lower bound will be calculated by underestimating f_1 within the considered box by a convex underestimator. In [14] the lower bound is calculated by interval arithmetic. Certainly, it is possible to replace f_1 by any $f_j,\ i \in \{1, \ldots, m\}$ or by a weighted sum of the objectives or similar. In [14] Fernández and Tóth use a similar termination rule to the following, where $F\colon \mathbb{R}^n \to \mathbb{R}^n$ is the natural interval extension of f, a common tool of interval arithmetic, see [35]:

Termination rule: *Store X^* in $\mathcal{L}_S$ if the following condition for given $\varepsilon, \delta > 0$ holds: $\omega(X^*) < \varepsilon$ and $\omega(F(X^*)) < \delta$.*

We use a much more elaborated termination procedure which we introduce in Section 3.3. Our modified termination rule guarantees the (ε, δ)-optimality of the calculated approximation set. For the discarding test several criteria have already been proposed in the literature. A first one is to use information on the monotonicity of the objective functions. Fernández and Tóth introduce in [14] a monotonicity test for biobjective optimization problems. A generalization to more than two objective functions can be found in [39]. Another class of discarding tests compares known objective values, which serve as upper bounds for the global minima, with lower bounds for the function values over the subboxes. We will follow this approach. Of course it is possible to combine the discarding test in order to accelerate the algorithm and use the different advantages of each test. But in the next sections we focus on the new ideas and test the algorithm only with our new approach.

3.1 Upper bounds and lower bounds by convex underestimators

We will generate a *stable* set $\mathcal{L}_{PNS}$ of objectives values (called *provisional nondominated solutions*) representing upper bounds for the global minima for (P). A set $\mathcal{N}$ of $\mathbb{R}^m$ is *stable* if for any $y^1, y^2 \in \mathcal{N}$ either $y^1 \not\leq y^2$ or $y^1 = y^2$ holds. Every time a point q is a new candidate for $\mathcal{L}_{PNS}$ we check if this point is dominated by any other point of $\mathcal{L}_{PNS}$. In this case q will not be included in $\mathcal{L}_{PNS}$. Otherwise, q will be added to $\mathcal{L}_{PNS}$ and all by q dominated points will be removed. Figure 1 shows an example for $\mathcal{L}_{PNS}$.

Having this list of upper bounds, a discarding test for a given box X^* requires also a lower bound for the values of f over X^*. Let $LB \subseteq \mathbb{R}^m$ be a set with $f(X^*) \subseteq LB + \mathbb{R}^m_+$.

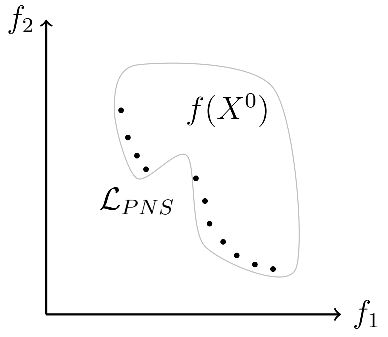


Figure 1: Example for $\mathcal{L}_{PNS},\ m = 2$.

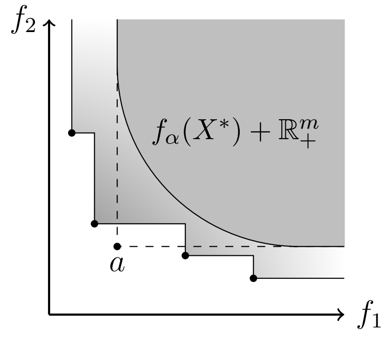


Figure 2: Upper image set $f_\alpha(X^*) + \mathbb{R}^m_+$ for $m = 2$, $\mathcal{L}_{PNS}$ and ideal point a of f_α.

If $LB \subseteq (\mathcal{L}_{PNS} + \mathbb{R}^m_+) \setminus \mathcal{L}_{PNS}$, then the box X^* can be discarded, because every point of $f(X^*)$ is in $(\mathcal{L}_{PNS} + \mathbb{R}^m_+) \setminus \mathcal{L}_{PNS}$. Hence every point of $f(X^*)$ is dominated by one point of $\mathcal{L}_{PNS}$. For the set LB we first recall the approach proposed so far in the literature (I) and then present our new approach which consists of two steps (II) and (III):

(I) proposed in [14]: LB is chosen as the lower bound of a box which contains all values of f on X^* and which is generated by the natural interval extension F of f on X^*, see [35].

(II) LB is chosen as the ideal point of the convex underestimators of the functions f_j over X^*. Recall that the *ideal point* of a MOP is determined componentwise by minimizing each function individually, i.e. we choose the point

$$a \in \mathbb{R}^m \text{ with } a_j := \min\{f_{j,\alpha}(x) \mid x \in X^*\} \quad \text{for} \quad j = 1, \ldots, m, \tag{2}$$

and set $LB = \{a = (a_1, \ldots, a_m)\}$. For an illustration see Figure 2.

(III) Find a tighter and not necessarily singleton set LB, i.e. a set LB with $f(X^*) \subseteq LB + \mathbb{R}^m_+$ and with $(LB + \mathbb{R}^m_+) \setminus (f(X^*) + \mathbb{R}^m_+)$ as small as possible by using convex underestimators and techniques from convex multiobjective optimization. We illustrate and discuss this new discarding test in Section 3.2.

We start by shortly discussing the first step of our new approach, i.e. we show that already a from (II), see (2), delivers a lower bound:

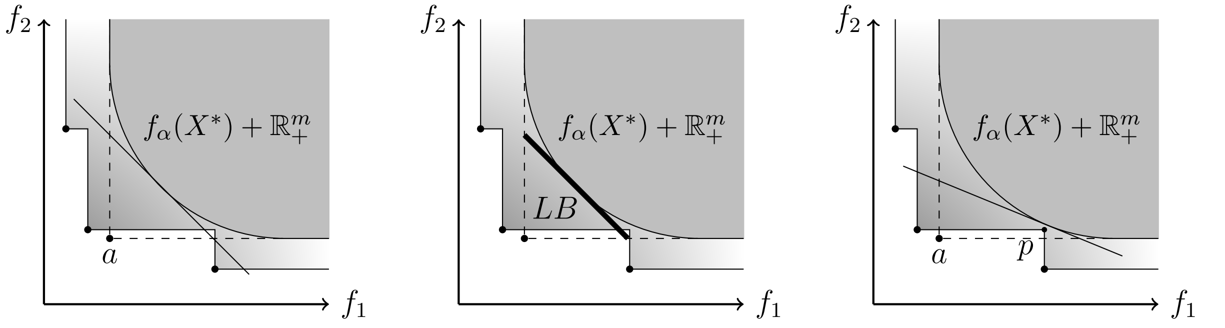
Lemma 3.1. *Let $f_{j,\alpha}$ be a convex underestimator of f_j on X^* for $j = 1, \ldots, m$ and define $a \in \mathbb{R}^m$ by (2). Then $a \leq f(x)$ for all $x \in X^*$, i.e. $f(X^*) \subseteq \{a\} + \mathbb{R}^m_+$.*

Proof. Let $j \in \{1, \ldots, m\}$. Because of $f_{j,\alpha}$ being a convex underestimator of f_j on X^* and the definition of a_j it follows $a_j \leq f_{j,\alpha}(x) \leq f_j(x)$ for all $x \in X^*$. □

Numerical experiments show that using (I) or (II) makes no big difference in their outcome. In some cases the lower bounds by interval arithmetic are better than the ones by convex underestimators and vice versa. However, (II) has the advantages the maximal error between the computed lower bound and actual images is bounded, see Remark 2.2 and Lemma 2.3. Moreover it is possible to improve (II) by using the convex underestimators, which is not possible for (I). This is the basic idea of our new discarding test.

3.2 Improved lower bounds by selected Benson cuts

The tighter bounds will be reached by adding cuts as known from Benson's outer approximation algorithm for convex vector optimization problems, see [3, 12] for Benson's algorithm. The cuts separate selected points p from the upper image set of the convex underestimators such that the cuts are supporting hyperplanes, see Figures 3a and 3c. This leads to new sets LB as can be seen in Figure 3b. As one can see, the cut in Figure 3a would lead to a set LB which does not allow to discard the box, while the cut in Figure 3c does. We explain first how to choose the points p such that we can generate cuts as in Figure 3c before we discuss how to calculate the cuts in more detail.


f_2
$f_\alpha(X^*) + \mathbb{R}^m_+$
a
f_1
f_2
$f_\alpha(X^*) + \mathbb{R}^m_+$
LB
f_1
f_2
$f_\alpha(X^*) + \mathbb{R}^m_+$
a
p
f_1

(a) A cut which separates a from $f_\alpha(X^*) + \mathbb{R}^m_+$.

(b) A lower bound LB for the situation in Figure 3a.

(c) A cut which separates p from $f_\alpha(X^*) + \mathbb{R}^m_+$.

Figure 3: Possible cuts to get a tighter set LB with $f(X^*) \subseteq LB + \mathbb{R}^m_+, m = 2$.

We use as points p so called local upper bounds. These are a versatile concept which is used in different fields of multiobjective optimization. Let a finite and stable list of function values $\mathcal{N} \subseteq f(X^0)$ be given. We call points in $\mathcal{N}$ also feasible points. Let $\hat{Z}$ be a box with $f(X^0) \subseteq \operatorname{int}(\hat{Z})$ (where int denotes the interior). The *search region* S is the set which contains all points which are not dominated by $\mathcal{N}$, i.e.

$$S := \{z \in \operatorname{int}(\hat{Z}) \mid \forall q \in \mathcal{N}, q \nleq z\} = \operatorname{int}(\hat{Z}) \setminus (\bigcup_{q\in\mathcal{N}} \{q\} + \mathbb{R}^m_+). \tag{3}$$

This set can be characterized with the help of *local upper bounds* which are the elements of the local upper bound set as defined below.

Definition 3.2. *[26] Let $\mathcal{N}$ be a finite and stable set of feasible points. A list $\mathcal{L} \subseteq \hat{Z}$ is called a local upper bound set with respect to $\mathcal{N}$ if*

(i) $\forall z \in S \; \exists p \in \mathcal{L} : z < p,$

(ii) $\forall z \in \left(\operatorname{int}(\hat{Z})\right) \setminus S \; \forall p \in \mathcal{L} : z \not< p$ *and*

(iii) $\forall p^1, p^2 \in \mathcal{L} : p^1 \nleq p^2$ *or* $p^1 = p^2$.

The next lemma gives an equivalent characterization, [26].

Lemma 3.3. *[26] A set $\mathcal{L}$ is called a local upper bound set with respect to a finite and stable set of feasible points $\mathcal{N}$ if and only if $\mathcal{L}$ consists of all points $p \in \hat{Z}$ that satisfy the following two conditions:*

(i) no point of $\mathcal{N}$ strictly dominates p and

(ii) for any $z \in \hat{Z}$ such that $z \geq p,\ z \neq p$, there exists $\bar{z} \in \mathcal{N}$ such that $\bar{z} < z$, i.e. p is a maximal point with property (i).

The following lemma is useful to understand the relations between $\mathcal{N}$ and $\mathcal{L}$ and will be important for our proof of Lemma 3.5. It is due to [25].

Lemma 3.4. *Let $\mathcal{L}$ be a local upper bound set with respect to a finite and stable set of feasible points $\mathcal{N}$. For every $\bar{z} \in \mathcal{N}$ and for every $j \in \{1, \ldots, m\}$ there is a $p \in \mathcal{L}$ with $\bar{z}_j = p_j$ and $\bar{z}_r < p_r$ for all $r \in \{1, \ldots, m\} \setminus \{j\}$.*

Proof. Let $\bar{z} \in \mathcal{N} \subseteq \operatorname{int}(\hat{Z})$. As $\mathcal{N}$ is a finite set, there exists a neighbourhood of $\bar{z}$ such that no other point of $\mathcal{N}$ is contained in that neighbourhood. Hence, let $\nu > 0$ such that $N_\nu(\bar{z}) := \{z \in \mathbb{R}^m \mid \|\bar{z} - z\| \leq \nu\} \subseteq \operatorname{int}(\hat{Z})$ and with $N_\nu(\bar{z}) \cap \mathcal{N} = \{\bar{z}\}$. Then we obtain for ν small enough $N_\nu(\bar{z}) \cap S = N_\nu(\bar{z}) \setminus (\{\bar{z}\} + \mathbb{R}^m_+)$. Now fix a $j \in \{1, \ldots, m\}$ and let $(\delta_t)_{t\in\mathbb{N}}$ be a null sequence with $\nu > \delta_t > 0$ for all $t \in \mathbb{N}$. Then we consider the sequence $(q^t)_{t\in\mathbb{N}}$ defined componentwise by: $q^t_j = \bar{z}_j - \delta_t$ and $q^t_r = \bar{z}_r$ for all $r \in \{1, \ldots, m\} \setminus \{j\}$ and for all $t \in \mathbb{N}$. Then $\lim_{t\to\infty} q^t = \bar{z}$ and $q^t \in N_\nu(\bar{z}) \cap S$ for all $t \in \mathbb{N}$. Hence with Definition 3.2 (i) we conclude that there is a local upper bound $p^t \in \mathcal{L}$ with $q^t < p^t$ for every $t \in \mathbb{N}$. Since $\mathcal{L}$ is also a finite set, the sequence $(p^t)_{t\in\mathbb{N}}$ contains a constant subsequence with a (limit) value $\bar{p}^j \in \mathcal{L}$. For its limit value it holds $\bar{z} \leq \bar{p}^j$. Moreover, for all $t \in \mathbb{N}$ of the constant subsequence we have $\bar{z}_r = q^t_r < p^t_r = \bar{p}^j_r$ for all $r \in \{1, \ldots, m\} \setminus \{j\}$. By Lemma 3.3 (i) it follows $\bar{z}_j = \bar{p}^j_j$. □

As a result of Lemma 3.4, for every $\bar{z} \in \mathcal{N}$ there exists a local upper bound $p \in \mathcal{L}$ with $\bar{z} \leq p$. In our context the list $\mathcal{L}_{PNS}$ is the list $\mathcal{N}$. The local upper bound set will be denoted by $\mathcal{L}_{LUB}$. An algorithm which calculates $\mathcal{L}_{LUB}$ w.r.t. $\mathcal{L}_{PNS}$ can be found in [26]. Figure 4 illustrates the sets $\mathcal{L}_{PNS}$ and $\mathcal{L}_{LUB}$. The values $\overline{M}_1$ and $\overline{M}_2$ in Figure 4 are upper bounds for the values of f_1 and f_2 in X^0, which do not have to be tight bounds and which can be easily computed for example with interval arithmetic.

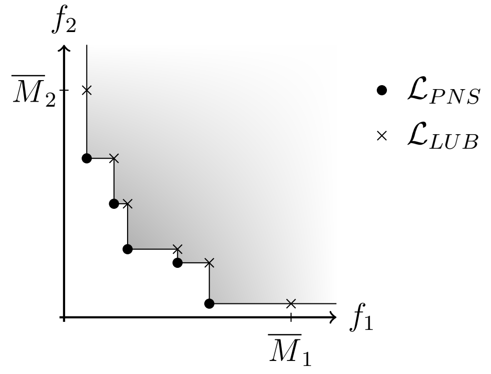


Figure 4: $\mathcal{L}_{PNS}$ and $\mathcal{L}_{LUB}$, cf. [26, Fig. 1]

The local upper bounds w.r.t. the set $\mathcal{L}_{PNS}$ are important as we can discard a box X^* if no local upper bound is contained in the set $f_\alpha(X^*) + \mathbb{R}^m_+$.

Lemma 3.5. *Let a box $X^* \subseteq X^0 \in \mathbb{R}^n$ be given and let $f_{j,\alpha}$ be a convex underestimator of f_j on X^* for $j = 1, \ldots, m$. Let $\mathcal{L}_{LUB} \subseteq \mathbb{R}^m$ be the local upper bound set w.r.t. $\mathcal{L}_{PNS}$. If*

$$\forall \bar{p} \in \mathcal{L}_{LUB} : \bar{p} \notin f_\alpha(X^*) + \mathbb{R}^m_+ \ . \tag{4}$$

then X^* does not contain any efficient point of (P).

Proof. Assume that there is some efficient point x^* of (P) with $x^* \in X^*$. Because of $f_{j,\alpha}$ being a convex underestimator for f_j on X^* for all $j \in \{1, \ldots, m\}$ all local upper bounds $\bar{p} \in \mathcal{L}_{LUB}$ do also not belong to $f(X^*) + \mathbb{R}^m_+$. Thus $f(x^*) \not\leq \bar{p}$ for all $\bar{p} \in \mathcal{L}_{LUB}$. From Definition 3.2 (i) it follows that $f(x^*)$ cannot be an element of S. With (3) we conclude that there exists a point $q \in \mathcal{L}_{PNS}$ with $q \leq f(x^*)$. As q is the image of a feasible point of (P) and as x^* is efficient for (P) we obtain $f(x^*) = q \in \mathcal{L}_{PNS}$. By Lemma 3.4 a local upper bound $p \in \mathcal{L}_{LUB}$ exists with $f(x^*) \leq p$ or $p \in \{f(x^*)\} + \mathbb{R}^m_+$. Again, as $f_{j,\alpha}$ are convex underestimators of f_j on X^* for all $j \in \{1, \ldots, m\}$ we conclude $p \in \{f_\alpha(x^*)\} + \mathbb{R}^m_+$ which is a contradiction to (4). Thus X^* contains no efficient point. □

Condition (4) can be tested by solving a scalar-valued convex optimization problem for every $p \in \mathcal{L}_{LUB}$. Instead of solving such an optimization problem for each point $\bar{p}$ we solve it for a few points and get immediately the information on how to generate and improve an outer approximation of $f_\alpha(X^*) + \mathbb{R}^m_+$. This information can then be used to efficiently reduce the number of points for which the scalar-valued problem has to be solved. This corresponds to generating tighter lower bounds for the values of f over a box. Thus let $\bar{p} \in \mathcal{L}_{LUB}$ and let a box X^* be given now. We check if $\bar{p} \in f_\alpha(X^*) + \mathbb{R}^m_+$ holds by solving the following scalar-valued convex optimization problem:

$$\min_{(x,t) \in \mathbb{R}^{n+1}} t \text{ s.t. } x \in X^*, \bar{p} + te \geq f_\alpha(x). \tag{$P_{\bar{p},X^*}$}$$

A minimal solution of $(P_{\bar{p},X^*})$ is named $(\tilde{x}, \tilde{t})$. If $\tilde{t} \leq 0$, then it holds $\bar{p} \in f_\alpha(X^*) + \mathbb{R}^m_+$. Otherwise, if $\tilde{t} > 0$, the point $\bar{p}$ lies outside and can be separated from $f_\alpha(X^*) + \mathbb{R}^m_+$ with a supporting hyperplane. This is used for our new discarding test.

The test is applied to a box X^* and consists of a finite number of iterations where an outer approximation of $f_\alpha(X^*) + \mathbb{R}^m_+$ is determined. The initial outer approximation is $\{a\} + \mathbb{R}^m_+$, where a is the ideal point of f_α on X^*, see (2). Recall that we can discard the box X^* if the following holds: In each iteration a local upper bound $\bar{p}$ is chosen. The first step is the comparison of the current outer approximation of $f_\alpha(X^*) + \mathbb{R}^m_+$ with $\bar{p}$ by checking if the inequalities which describe the outer approximation are satisfied. In case $\bar{p}$ is not an element of this outer approximation, the next iteration, namely choosing a next local upper bound, starts. Otherwise we continue with solving $(P_{\bar{p},X^*})$ to get now the position of $\bar{p}$ w.r.t. $f_\alpha(X^*) + \mathbb{R}^m_+$. The above mentioned cases ($\tilde{t} > 0$, $\tilde{t} \leq 0$) can occur. We extend this to three cases to reduce the effort as only ε-optimality is the aim for a given scalar $\varepsilon > 0$. Hence we differentiate between the following cases for the minimum value $\tilde{t}$ of $(P_{\bar{p},X^*})$:

(1) $\tilde{t} \leq 0$, i.e. $\bar{p} \in f_\alpha(X^*) + \mathbb{R}^m_+$: Efficient points in X^* are possible. Thus X^* cannot be discarded and we distinguish between the two subcases:

(1a) $\tilde{t} < -\frac{\varepsilon}{2}$: Stop the whole discarding test in order to bisect X^* later.

(1b) $-\frac{\varepsilon}{2} \leq \tilde{t} \leq 0$: Construct a supporting hyperplane to improve the outer approximation of $f_\alpha(X^*) + \mathbb{R}^m_+$ and choose the next local upper bound. Moreover set a flag that X^* cannot be discarded.

(2) $\tilde{t} > 0$, i.e. $\bar{p} \notin f_\alpha(X^*) + \mathbb{R}^m_+$: Construct a supporting hyperplane to improve the outer approximation of $f_\alpha(X^*) + \mathbb{R}^m_+$ and choose the next local upper bound.

Only if the last case holds for every local upper bound, the box X^* can be discarded, see Lemma 3.5. Case (1b) is motivated by getting ε-optimal points at the end of the algorithm as we will prove later. To store X^* in the solution list, there has to be at least one $\bar{p}$ where the case (1b) is fulfilled and no $\tilde{p}$ with (1a).

During the discarding test new supporting hyperplanes are constructed, if $\tilde{t} \geq -\frac{\varepsilon}{2}$. The support vector of such a hyperplane is $\tilde{y} := \bar{p} + \tilde{t}e$. For calculating a normal vector $\lambda^* \in \mathbb{R}^m$ of the supporting hyperplane a procedure is given in [12], where a scalar-valued linear optimization problem has to be solved. In difference to this and with help of properties of duality theory we can also use a Lagrange multiplier $\lambda^* \in \mathbb{R}^m$ to the constraint $\bar{p} + te \geq f_\alpha(x)$ to get a normal vector of the supporting hyperplane, as explained in detail in [30]. The following algorithm describes the procedure, where the flag $\mathcal{D}$ stands for the decision to discard a box after the algorithm and the flag $\mathcal{B}$ for bisecting the box, respectively.

Algorithm 2 Discarding test

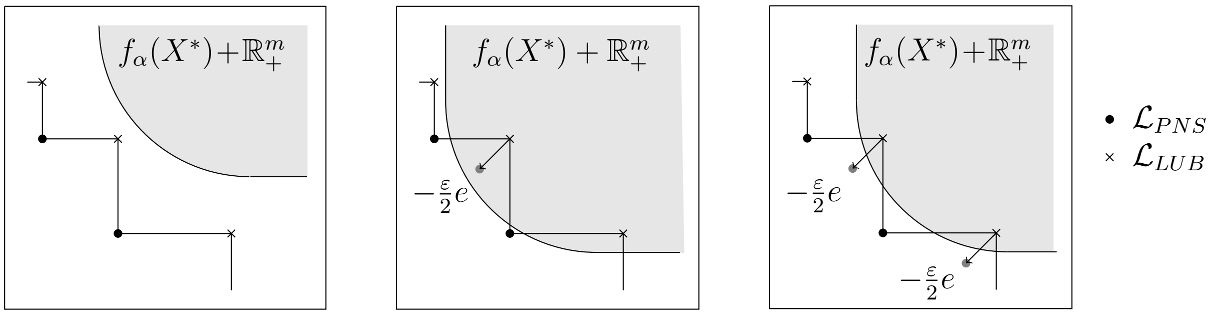
```
INPUT: $X^* \in \mathbb{R}^n,\ f \in \mathcal{C}^2(\mathbb{R}^n, \mathbb{R}^m),\ \mathcal{L}_{PNS}, \mathcal{L}_{LUB} \subseteq \mathbb{R}^m, \varepsilon > 0, \alpha$
OUTPUT: flags $\mathcal{D}, \mathcal{B}$, lists $\mathcal{L}_{PNS}, \mathcal{L}_{LUB}$
 1: Compute for every objective function its convex underestimator on $X^*$ and its corresponding minimum $x^j$, update $\mathcal{L}_{PNS}$ by $f(x^j)$ and $\mathcal{L}_{LUB} =: \{p^1, \ldots, p^k\}$
 2: $\mathcal{D} \leftarrow 1,\ \mathcal{B} \leftarrow 0$
 3: for $s = 1, \ldots, k$ do
 4:     if $p^s$ is inside the current outer approximation of $f_\alpha(X^*) + \mathbb{R}^m_+$ then
 5:         Solve $(P_{p^s, X^*})$ with solution $(\tilde{x}, \tilde{t})$
 6:         if $\tilde{t} < -\frac{\varepsilon}{2}$ then
 7:             break for-loop
 8:             Set flags $\mathcal{D} \leftarrow 0$ and $\mathcal{B} \leftarrow 1$
 9:         else if $\tilde{t} \leq 0$ then
10:             Update outer approximation of $f_\alpha(X^*) + \mathbb{R}^m_+$ and set flag $\mathcal{D} \leftarrow 0$
11:         else Update outer approximation of $f_\alpha(X^*) + \mathbb{R}^m_+$
```

Actually we could add in line 1 the image of any other point of X^0 for example the midpoint of the considered box X^* to the list $\mathcal{L}_{PNS}$, because we only have box constraints. But as we plan to work also with constraints later the feasibility of the pre-images of all points of $\mathcal{L}_{PNS}$ has to be ensured. Thus we need that $\mathcal{L}_{PNS} \subseteq f(M) \subseteq f(X^0)$, which is the case for our construction in line 1. For more information about the handling of constraints we refer to Section 5. Note that the condition of line 4 can be checked by evaluating a finite number of inequalities which are given by the current outer approximation. The following theorem gives the correctness of this discarding test.

Theorem 3.6. *Let a box $X^* \subseteq X^0 \in \mathbb{R}^n$ and (P) be given. Let $\mathcal{L}_{LUB} \subseteq \mathbb{R}^m$ be a local upper bound set w.r.t. $\mathcal{L}_{PNS}$. If X^* contains an efficient point of (P), then the output of Algorithm 2 is $\mathcal{D} = 0$, i.e. X^* will not be discarded by Algorithm 2.*

Proof. Assume that there is some efficient point x^* of (P) with $x^* \in X^*$. Suppose the output of Algorithm 2 applied to X^* is $\mathcal{D} = 1$. This means that for all local upper bounds either the conditions in lines 6 and 9 are not satisfied or they are not contained in the current outer approximation (see line 4). In case the latter occurs the local upper bound is clearly not in $f_\alpha(X^*) + \mathbb{R}^m_+$. For the other local upper bounds $\bar{p} \in \mathcal{L}_{LUB}$ which are not satisfying lines 6 and 9, but line 4 the optimization problem $(P_{\bar{p},X^*})$ has also a minimal value $\tilde{t} > 0$. Hence it holds for all $\bar{p} \in \mathcal{L}_{LUB}$ that $\bar{p} \notin f_\alpha(X^*) + \mathbb{R}^m_+$ and with Lemma 3.5 we have a contradiction to X^* contains an efficient point. $\square$

All possibilities of the results of the discarding test applied to a box X^* in the two-dimensional case are visualized in Figures 5a to 5c.



$f_\alpha(X^*)+\mathbb{R}^m_+$
$f_\alpha(X^*)+\mathbb{R}^m_+$
$-\frac{\varepsilon}{2}e$
$f_\alpha(X^*)+\mathbb{R}^m_+$
$-\frac{\varepsilon}{2}e$
$-\frac{\varepsilon}{2}e$
• $\mathcal{L}_{PNS}$
× $\mathcal{L}_{LUB}$

(a) Discard box X^*.

(b) Do not discard box X^*, but bisect it.

(c) Neither discard box X^*, nor bisect it.

Figure 5: Possible situations during a discarding test applied to box X^*.

The list $\mathcal{L}_{PNS}$ is growing during the algorithm. Hence we get better upper bounds for the nondominated solutions. It is possible that a box X^* was not discarded and not bisected by Algorithm 2 while it could be discarded if we would compare it with the final list $\mathcal{L}_{PNS}$. Thus we add an additional test to the algorithm after the first `while`-loop of executions of Algorithm 2 to detect such boxes and discard them. In that case the lists $\mathcal{L}_{PNS}$ and $\mathcal{L}_{LUB}$ are static and will not be updated as in line 1 of Algorithm 2. This method can be found in Algorithm 3. Additionally, in that procedure we re-use the already calculated approximation of $f_\alpha(X^*) + \mathbb{R}^m_+$ and, what is more, we save the feasible points $\tilde{x}$ which are a solution of $(P_{\bar{p},X^*})$ if its corresponding $\tilde{t}$ is between $-\frac{\varepsilon}{2}$ and 0. Note that the case $\tilde{t} < -\frac{\varepsilon}{2}$ is not possible for a box X^*, which passed Algorithm 2 with any bisecting $(\mathcal{B} = 1)$. This fact will be shown in Lemma 4.5. The points $\tilde{x}$ with $-\frac{\varepsilon}{2} \leq \tilde{t} \leq 0$ are collected in the list $\mathcal{X}$ and will serve as the possible points of the (ε, δ)-minimal set $\mathcal{A}$.

Algorithm 3 Discarding test with static lists $\mathcal{L}_{PNS}, \mathcal{L}_{LUB}$

```
INPUT: $X^* \in \mathbb{R}^n,\ f \in \mathcal{C}^2(\mathbb{R}^n, \mathbb{R}^m),\ \mathcal{L}_{PNS}, \mathcal{L}_{LUB} = \{p^1, \dots, p^k\} \subseteq \mathbb{R}^m, \varepsilon > 0, \alpha$
OUTPUT: list $\mathcal{X}$, flag $\mathcal{D}$
1: if there is no current outer approximation of $f_\alpha(X^*) + \mathbb{R}^m_+$ calculated yet then
2:     Calculate ideal point $a$ of $f_\alpha$ and initialize $\{a\} + \mathbb{R}^m_+$ as an outer approximation of $f_\alpha(X^*) + \mathbb{R}^m_+$
3: $\mathcal{D} \leftarrow 1$, $\mathcal{X} \leftarrow \emptyset$
4: for $s = 1, \dots, k$ do
5:     if $p^s$ is inside the current outer approximation of $f_\alpha(X^*) + \mathbb{R}^m_+$ then
6:         Solve $(P_{p^s, X^*})$ with solution $(\tilde{x}, \tilde{t})$
7:         Update outer approximation of $f_\alpha(X^*) + \mathbb{R}^m_+$
8:         if $\tilde{t} \leq 0$ then $\mathcal{X} \leftarrow \mathcal{X} \cup \tilde{x}$ and set flag $\mathcal{D} \leftarrow 0$
```

Theorem 3.7. *Let a box $X^* \subseteq X^0 \in \mathbb{R}^n$ and (P) be given. Let $\mathcal{L}_{LUB} \subseteq \mathbb{R}^m$ be a local upper bound set w.r.t. $\mathcal{L}_{PNS}$. If X^* contains an efficient point of (P), then the output of Algorithm 2 is $\mathcal{D} = 0$, i.e. X^* will not be discarded by Algorithm 3.*

Proof. The proof is analogue to the proof of Theorem 3.6. □

3.3 The Complete Algorithm

Having now the new discarding test we can present the complete algorithm together with the new termination procedure. The whole algorithm for (P) is the following:

Algorithm 4 Algorithm to find an (ε, δ)-set of (P)

```
INPUT: $X^0 \in \mathbb{R}^n, f \in \mathcal{C}^2(\mathbb{R}^n, \mathbb{R}^m), \varepsilon > 0, \delta > 0$
OUTPUT: $\mathcal{A}, \mathcal{L}_{S,3}, \mathcal{L}_{PNS}, \mathcal{L}_{LUB}$
 1: $\mathcal{L}_W \leftarrow \{X^0\}, \mathcal{L}_{S,1} \leftarrow \emptyset, \mathcal{L}_{S,2} \leftarrow \emptyset, \mathcal{L}_{S,3} \leftarrow \emptyset, A \leftarrow \emptyset, \mathcal{L}_{PNS} \leftarrow \emptyset, \mathcal{L}_{LUB} \leftarrow \emptyset$
 2: Calculate $\alpha$ such that $f_{j,\alpha}$ is a convex underestimator of $f_j$ on $X^0,\ j = 1, \dots, m$
 3: while $\mathcal{L}_W \neq \emptyset$ do
 4:     Select a box $X^*$ from $\mathcal{L}_W$ and delete it from $\mathcal{L}_W$
 5:     Bisect $X^*$ perpendicularly to a direction of maximum width $\rightarrow X^1, X^2$
 6:     for $l = 1, 2$ do
 7:         Apply Algorithm 2 to $X^l$
 8:         if $\mathcal{B} = 1$ then Store $X^l$ in $\mathcal{L}_W$
 9:         else if $\mathcal{D} = 0$ then Store $X^l$ in $\mathcal{L}_{S,1}$
10:         else Discard $X^l$
11: while $\mathcal{L}_{S,1} \neq \emptyset$ do
12:     Select a box $X^*$ from $\mathcal{L}_{S,1}$ and delete it from $\mathcal{L}_{S,1}$
13:     Apply Algorithm 3 to $X^*$
14:     if $\mathcal{D} = 0$ then Store $X^*$ in $\mathcal{L}_{S,2}$
15:     else Discard $X^*$
16: while $\mathcal{L}_{S,2} \neq \emptyset$ do
17:     Select a box $X^*$ from $\mathcal{L}_{S,2}$ and delete it from $\mathcal{L}_{S,2}$
18:     Apply Algorithm 3 to $X^*$ and obtain $\mathcal{X}$
19:     if $\mathcal{D} = 1$ then Discard $X^*$
20:     else if $\mathcal{D} = 0$ and $\omega(X^*) \leq \delta$ then
21:         $\{x^1, \dots, x^k\} \leftarrow \mathcal{X}$
22:         for $s = 1, \dots, k$ do
23:             if $f(x^s) \leq \bar{p}$ for at least one $\bar{p} \in \mathcal{L}_{LUB}$ or $\omega(X^*) < \sqrt{\frac{\varepsilon}{\alpha}}$ then
24:                 $\mathcal{A} \leftarrow \mathcal{A} \cup \{x^s\}$
25:                 Store $X^*$ in $\mathcal{L}_{S,3}$ afterwards
26:     if $\mathcal{D} = 0$ and no point of $\mathcal{X}$ was stored in $\mathcal{A}$ then
27:         Bisect $X^*$ perpendicularly to a direction of maximum width $\rightarrow X^1, X^2$
28:         Store $X^1$ and $X^2$ in $\mathcal{L}_{S,2}$
29: $\mathcal{A} \leftarrow \mathcal{A} \cup \{x \in X^0 \mid f(x) \in \mathcal{L}_{PNS}\}$
```

The algorithm consists of three `while`-loops. The list $\mathcal{L}_{S,t},\ t = 1, 2, 3$ is the solution list of the t-th `while`-loop and becomes the working list for the next loop if $t = 1, 2$. The first loop from line 3 handles the basic discarding test, which was explained before in detail. It generates the list $\mathcal{L}_{PNS}$ until this lists is near to the nondominated set in dependence to ε. All boxes X from $\mathcal{L}_{S,1}$ have the following properties:

$$\exists \bar{p} \in \mathcal{L}_{LUB} : \bar{p} \in f_\alpha(X) + \mathbb{R}^m_+ \tag{5}$$

$$\forall p \in \mathcal{L}_{LUB} : p - \frac{\varepsilon}{2} e \notin f_\alpha(X) + \mathbb{R}^m_+ \tag{6}$$

The first property is true, because if all local upper bounds would be outside of $f_\alpha(X) + \mathbb{R}^m_+$, then the box X would be discarded, see Lemma 3.5 for the proof. If the second property does not hold for X, this box would not have been stored in the solution list $\mathcal{L}_{S,1}$, because line 6 of Algorithm 2 would be satisfied and X would be bisected into its subboxes.

With the second and third `while`-loop of the algorithm we do not loose these characteristics (5) and (6). The second loop only checks whether some boxes from $\mathcal{L}_{S,1}$ can be discarded by the final lists $\mathcal{L}_{PNS}$ and $\mathcal{L}_{LUB}$. In the third loop we check for every box $X \in \mathcal{L}_{S,2}$ if $\omega(X)$ is less than or equal to δ for a predefined $\delta > 0$. If the box X is not small enough we bisect it and apply the discarding test to both subboxes. But if $\omega(X) \leq \delta$ we add those elements of $\mathcal{X}$, which images are less than or equal to a local upper bound, to the set $\mathcal{A}$. Furthermore if $\omega(X)$ is bounded by $\sqrt{\frac{\varepsilon}{\alpha}}$ (usually smaller than δ) we add all elements of $\mathcal{X}$ to $\mathcal{A}$. Note that $\mathcal{X} \neq 0$ if and only if $\mathcal{D} = 1$. Moreover the pre-images of the points of the final list $\mathcal{L}_{PNS}$ are also added to $\mathcal{A}$ at the end of the algorithm. Summarized we have

$$\begin{aligned}\mathcal{A} &:= \{x \in X^0 \mid f(x) \in \mathcal{L}_{PNS}\} \\ &\cup \bigcup_{X^* \in \mathcal{L}_{S,2}} \left\{ x \in X^* \,\middle|\, \begin{array}{c} \exists \bar{p} \in \mathcal{L}_{LUB}\ \exists t \leq 0 : \\ (x,t) \text{ is a minimal solution of } (P_{\bar{p},X^*}) : \\ (\omega(X^*) \leq \delta \text{ and } \exists \tilde{p} \in \mathcal{L}_{LUB} : f(x) \leq \tilde{p}) \text{ or } \omega(X^*) < \sqrt{\frac{\varepsilon}{\alpha}} \end{array} \right\}\end{aligned}$$

Note that the union is not a disjoint union. The fact that $\mathcal{A}$ is an (ε, δ)-minimal set of (P), will be shown in Section 4.2.

4 Convergence Results

In this section we show the correctness and finiteness of Algorithm 4.

4.1 Termination

To show the termination of the algorithm we have to verify that each `while`-loop of Algorithm 4 is finite. We start with showing the termination of the first `while`-loop.

Lemma 4.1. *The first* `while`*-loop (line 3-10) of Algorithm 4 terminates.*

Proof. Assume the first `while`-loop does not terminate. Hence there must be an infinite sequence of boxes $X^0 \subset X^1 \subset \ldots \subset X^k \subset \ldots$ which were not discarded, but bisected after applying Algorithm 2 on each box. Hence every box X^k will be stored in $\mathcal{L}_W$ and bisected in another iteration, where the box X^{k+1} is one of the two obtained subboxes. Obviously the boxwidth decreases among the sequence of boxes, i.e. $\omega(X^k) \geq \omega(X^{k+1})$ for every $k \in \mathbb{N}$ and converges to 0 (because we divide the boxes perpendicular to a side with maximal width). Now we define δ_{X^0} like in (1). Lets choose the first box $X^{\tilde{k}}$ with $\omega(X^{\tilde{k}}) \leq \delta_{X^0}$. From Lemma 2.3 for every objective function it follows for all $x \in X^{\tilde{k}} : |f_j(x) - a_j| \leq \frac{\varepsilon}{2}$ for all $j = 1, \ldots, m$ or

$$f(x) \in (\{a\} + \mathbb{R}^m_+) \cap (\{a + \frac{\varepsilon}{2}c\} - \mathbb{R}^m_+). \tag{7}$$

Now consider the minima of each convex underestimator. Choose an arbitrary $\tilde{x}^j \in \operatorname{argmin}\{f_{j,\alpha}(x) \mid x \in X^{\tilde{k}}\}$ for every $j = 1, \ldots, m$. The images of the points $\tilde{x}_j$ under the original function, i.e. $f(\tilde{x}^j)$, are potential points of the list $\mathcal{L}_{PNS}$, see line 1 of Algorithm 2, and clearly satisfy (7) as well. Lets choose one of these and denote it with q. This point will be added to $\mathcal{L}_{PNS}$ if there is no other point from the current list $\mathcal{L}_{PNS}$ dominating q.

Because of the assumption that $X^{\tilde{k}}$ will be bisected, there must be a local upper bound $\bar{p}$ with the solution $(\tilde{x},\tilde{t})$ of $(P_{\bar{p},X^{\tilde{k}}})$ where $\tilde{t}$ is less than $-\frac{\varepsilon}{2}$. Now we want to check for each local upper bound if this is possible. If we show that for any local upper bound $\bar{p}$ we have that the solution of $(P_{\bar{p},X^{\tilde{k}}})$ is never less than $-\frac{\varepsilon}{2}$, we can thus conclude that the assumption is wrong and $X^{\tilde{k}}$ will not be bisected.

First consider all local upper bounds which do not belong to $\{a\}+\mathbb{R}^m_+$. Hence let $\bar{p}\in\mathcal{L}_{LUB}\setminus(\{a\}+\mathbb{R}^m_+)$, i.e. there is an $u\in\{1,\dots,m\}$ with $\bar{p}_u<a_u$. The condition in line 4 of Algorithm 2 is is not satisfied and $(P_{\bar{p},X^{\tilde{k}}})$ will not be solved.

Next we consider those $\bar{p}\in\mathcal{L}_{LUB}$ with $\bar{p}\in\mathcal{L}_{LUB}\cap(\{a\}+\mathbb{R}^m_+)$, in case there are any, and distinguish two cases: The first one is:

$$|\bar{p}_u-a_u|=\bar{p}_u-a_u\leq\frac{\varepsilon}{2}\text{ for one }u\in\{1,\dots,m\}. \tag{8}$$

Problem $(P_{\bar{p},X^{\tilde{k}}})$ is solved and has a solution $(\tilde{x},\tilde{t})$. Then it holds $\bar{p}_u+\tilde{t}\geq f_{u,\alpha}(\tilde{x})\geq a_u$. Hence $\frac{\varepsilon}{2}\geq\bar{p}_u-a_u\geq-\tilde{t}$, which leads to $\tilde{t}\geq-\frac{\varepsilon}{2}$. The second case is that there is some $\bar{p}\in\mathcal{L}_{LUB}\cap(\{a\}+\mathbb{R}^m_+)$ with

$$|\bar{p}_j-a_j|=\bar{p}_j-a_j>\frac{\varepsilon}{2}\text{ for all }j\in\{1,\dots,m\} \tag{9}$$

or equivalently $\bar{p}\in\{a+\frac{\varepsilon}{2}e\}+\operatorname{int}(\mathbb{R}^m_+)$. It follows for every $j\in\{1,\dots,m\}$: $a_j+\frac{\varepsilon}{2}<\bar{p}_j$, but we know there is a point q, see above, which is a feasible image of f on $X^{\tilde{k}}$, a candidate for $\mathcal{L}_{PNS}$ and belongs to the set $(\{a\}+\mathbb{R}^m_+)\cap(\{a+\frac{\varepsilon}{2}e\}-\mathbb{R}^m_+)$. Define

$$y:=\begin{cases}q' & \text{, if there is a } q'\in\mathcal{L}_{PNS}\text{ with }q'\leq q\\ q & \text{, otherwise.}\end{cases}$$

and thus $y\in\mathcal{L}_{PNS}$. Now we obtain by considering each component of a,y and $\bar{p}$ $y_j\leq q_j\leq a_j+\frac{\varepsilon}{2}<\bar{p}_j$ for all $j=1,\dots,m$. Hence y strictly dominates $\bar{p}$, which is a contradiction to Lemma 3.3 (i). Thus the existence of a local upper bound in $\{a+\frac{\varepsilon}{2}e\}+\operatorname{int}(\mathbb{R}^m_+)$ is not possible.

Clearly, it is not possible for $X^{\tilde{k}}$ that it satisfies the conditions for bisection. Hence the assumed infinite sequence of subboxes does not exist. Accordingly to this the first `while`-loop will terminate. □

Lemma 4.2. *The second `while`-loop (line 11-15) of Algorithm 4 terminates.*

Proof. The termination of the second `while`-loop is clear, because it has exactly $|\mathcal{L}_{S,1}|$ iterations. □

Lemma 4.3. *The third `while`-loop (line 16-28) of Algorithm 4 terminates.*

Proof. Assume the third `while`-loop does not terminate. Hence there must be an infinite sequence of boxes $X^1\subset\ldots\subset X^k\subset\ldots$ with $X^1\in\mathcal{L}_{S,2}$ after the second loop, which were not discarded, but bisected after applying Algorithm 3 on each box. Hence every box X^k will be stored in $\mathcal{L}_{S,2}$ and bisected in another iteration, where the box X^{k+1} is one of the two obtained subboxes. Obviously the boxwidth decreases among the sequence of boxes, i.e. $\omega(X^k)\geq\omega(X^{k+1})$ for every $k\in\mathbb{N}$ and converges to 0 (because we divide the boxes perpendicular to a side with maximal width). Lets choose the first box $X^{\tilde{k}}$ with

$\omega(X^{\tilde{k}}) < \min\{\delta, \sqrt{\frac{\varepsilon}{\alpha}}\}$. For $X^{\tilde{k}}$ we have $\mathcal{D} = 0$, otherwise it will be discarded. Therefore the conditions in lines 20 and 23 of Algorithm 4 are satisfied and $X^{\tilde{k}}$ will be stored in $\mathcal{L}_{S,3}$. This contradicts the assumption that $X^{\tilde{k}}$ will be bisected. Hence the assumed infinite sequence of subboxes does not exist. Accordingly to this the third `while`-loop will terminate. □

With these lemmas we obtain that the whole algorithm terminates.

4.2 Correctness

First we state that all efficient points x of (P) are contained in the union of boxes from the final list $\mathcal{L}_{S,3}$:

Lemma 4.4. *Let $\mathcal{L}_{S,3}$ be the output of Algorithm 4 for arbitrary $\varepsilon, \delta > 0$ and let $\mathcal{L}_{S,1}$ and $\mathcal{L}_{S,2}$ be the lists after the first and the second `while`-loop, respectively. Then*

$$X_E \subseteq \bigcup_{X \in \mathcal{L}_{S,3}} X \subseteq \bigcup_{X \in \mathcal{L}_{S,2}} X \subseteq \bigcup_{X \in \mathcal{L}_{S,1}} X.$$

Proof. This is a direct consequence of Theorems 3.6 and 3.7 and the way the lists are constructed. □

The next two lemmas show the in Section 3.2 on page 12 mentioned fact that in Algorithm 3 the case $\tilde{t} < -\frac{\varepsilon}{2}$ is not possible in the second and third `while`-loop of Algorithm 4.

Lemma 4.5. *Let $X \in \mathbb{IR}^m$ be chosen from the working list $\mathcal{L}_{S,1}$ during the second `while`-loop of Algorithm 4 and hence be an input for Algorithm 3. If $(P_{\bar{p},X})$ is solved for any $\bar{p} \in \mathcal{L}_{LUB}$ within Algorithm 3 we obtain a solution $(\tilde{x}, \tilde{t})$ with $\tilde{t} \geq -\frac{\varepsilon}{2}$.*

Proof. Let $\bar{p} \in \mathcal{L}_{LUB}$ be inside the current outer approximation of $f_\alpha(X) + \mathbb{R}^m_+$ and $(\tilde{x}, \tilde{t})$ the solution of $(P_{\bar{p},X})$. Assume now $\tilde{t} < -\frac{\varepsilon}{2}$. In particular by $f_\alpha(\tilde{x}) \leq \bar{p} + \tilde{t}e < \bar{p}$ we obtain $\bar{p} \in f_\alpha(X) + \mathbb{R}^m_+$. Because of $X \in \mathcal{L}_{S,1}$ this box was not discarded in the first `while`-loop of Algorithm 4, i.e. $\mathcal{D} = 0$ and $\mathcal{B} = 0$. For the next steps we consider X during the first `while`-loop where Algorithm 2 is executed. Let $\mathcal{L}'_{LUB}$ be the set of local upper bounds at this time. Then it holds

$$\forall p' \in \mathcal{L}'_{LUB} : \ (P_{p',X}) \text{ was solved with solution } (x', t') \Rightarrow t' \geq -\frac{\varepsilon}{2}. \tag{10}$$

Now we distinguish two cases: The first case is $\bar{p} \in \mathcal{L}'_{LUB}$. Because of $\bar{p} \in f_\alpha(X) + \mathbb{R}^m_+$ the problem $(P_{\bar{p},X})$ was solved. As $(\tilde{x}, \tilde{t})$ is feasible for $(P_{\bar{p},X})$ with $\tilde{t} < -\frac{\varepsilon}{2}$ this contradicts (10).

The second case is $\bar{p} \notin \mathcal{L}'_{LUB}$, i.e. $\bar{p}$ was added to the set of local upper bounds after considering X in the first `while`-loop. Then there exists a $p^* \in \mathcal{L}'_{LUB}$ with $\bar{p} \leq p^*$ and $\bar{p} \neq p^*$. This fact can be easily seen by induction over a generating algorithm of a local upper bound set, for example Algorithm 3 in [26]. Because of $\bar{p} \in f_\alpha(X) + \mathbb{R}^m_+$ it also holds $p^* \in f_\alpha(X) + \mathbb{R}^m_+$ and the optimization problem $(P_{p^*,X})$ was solved in Algorithm 2. Then $(\tilde{x}, \tilde{t})$ is also feasible for $(P_{p^*,X})$, because $f_\alpha(\tilde{x}) \leq \bar{p} + \tilde{t}e \leq p^* + \tilde{t}e$, which contradicts (10). □

Lemma 4.6. *Let $X \in \mathbb{IR}^m$ be chosen from the working list in $\mathcal{L}_{S,2}$ during the third `while`-loop of Algorithm 4 and hence be an input for Algorithm 3. If $(P_{\bar{p},X})$ is solved for any $\bar{p} \in \mathcal{L}_{LUB}$ within Algorithm 3 we obtain a solution $(\tilde{x}, \tilde{t})$ with $\tilde{t} \geq -\frac{\varepsilon}{2}$.*

Proof. Let $\bar{p} \in \mathcal{L}_{LUB}$ be inside the current outer approximation of $f_\alpha(X) + \mathbb{R}^m_+$ and $(\tilde{x}, \tilde{t})$ the solution of $(P_{\bar{p},X})$. Assume now $\tilde{t} < -\frac{\varepsilon}{2}$. In particular by $f_\alpha(\tilde{x}) \leq \bar{p} + \tilde{t}e < \bar{p}$ we obtain $\bar{p} \in f_\alpha(X) + \mathbb{R}^m_+$. Note that the set $\mathcal{L}_{LUB}$ is fixed after the first loop.

Because of Lemma 4.5 X was not considered in the second `while`-loop, i.e. $X \notin \mathcal{L}_{S,2}$ after line 15. But here exists a box X^* with $X \subseteq X^*$, which has been considered and not discarded in this loop. Let f^*_α be the componentwise convex underestimator of f on $X^* = [\underline{x^*}, \overline{x^*}]$, i.e. $f^*_{j,\alpha}(x) = f(x) - \frac{\alpha}{2}(\underline{x^*} - x)^T(\overline{x^*} - x)$ for all $x \in X^*, j = 1, \ldots, m$, which is clearly less than f_α on X, i.e. $f^*_\alpha(x) \leq f_\alpha(x)$ for all $x \in X$. Because of $\bar{p} \in f_\alpha(X) + \mathbb{R}^m_+$ it also holds $\bar{p} \in f^*_\alpha(X^*) + \mathbb{R}^m_+$ and the optimization problem $(P_{\bar{p},X^*})$ was solved in Algorithm 3. Let (x^*, t^*) be a solution of $(P_{\bar{p},X^*})$. Because of Lemma 4.5 we have $t^* \geq -\frac{\varepsilon}{2}$. Since $X \subseteq X^*$ and $f_\alpha(\tilde{x}) \geq f^*_\alpha(\tilde{x})$ the pair $(\tilde{x}, \tilde{t})$ is also feasible for $(P_{\bar{p},X^*})$, which contradicts the minimality of (x^*, t^*). □

Algorithm 4 calculates based on the list $\mathcal{L}_{LUB}$ and thus on the list $\mathcal{L}_{PNS}$ a tube, which contains all nondominated points of (P) and has width $\varepsilon/2$. We prove this in Theorem 4.8. Thereby the tube is defined as follows:

$$T \quad := \quad \left(\bigcup_{\bar{p} \in \mathcal{L}_{LUB}} \{\bar{p}\} - \mathbb{R}^m_+ \right) \setminus \left(\bigcup_{\bar{p} \in \mathcal{L}_{LUB}} \{\bar{p} - \tfrac{\varepsilon}{2} e\} - \operatorname{int}(\mathbb{R}^m_+) \right). \tag{11}$$

An illustration of such a tube T is shown in Figure 6. Recall that $\hat{Z}$ was defined as a box with $f(X^0) \subseteq \operatorname{int}(\hat{Z})$.

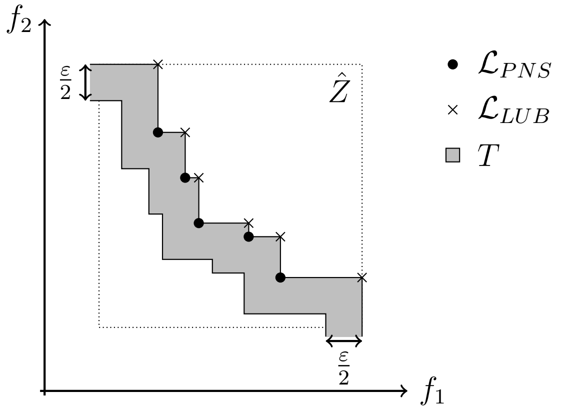


Figure 6: $\frac{\varepsilon}{2}$−tube T; $m = 2$.

The tube T contains all points from $\mathcal{L}_{PNS}$ which will be shown in the next lemma.

Lemma 4.7. *Let $\mathcal{L}_{LUB}$ be the local upper bound set w.r.t. $\mathcal{L}_{PNS}$. Let T be defined as in* (11)*. Then $\mathcal{L}_{PNS} \subseteq T$.*

Proof. Because of Lemma 3.4 there exists for every point $q \in \mathcal{L}_{PNS}$ a point $p^q \in \mathcal{L}_{LUB}$ with $q \in \{p^q\} - \mathbb{R}^m_+$. If q would belong to a set $\{p - \frac{\varepsilon}{2}e\} - \operatorname{int}(\mathbb{R}^m_+)$ for any $p \in \mathcal{L}_{LUB}$, then it would hold for every $j \in \{1, \ldots, m\} : q_j < p_j - \frac{\varepsilon}{2} < p_j$. But this contradicts Lemma 3.3 (i). Thus $\mathcal{L}_{PNS} \subseteq T$. □

Theorem 4.8. *Let $\mathcal{L}_{LUB}$ be the local upper bound set w.r.t. $\mathcal{L}_{PNS}$ with $\mathcal{L}_{PNS}$ from Algorithm 4 and let T be defined as in* (11)*. Let $\bar{x}$ be an efficient point of (P). Then $f(\bar{x}) \in T$, i.e. the nondominated set of (P) is a subset of T.*

Proof. Let $\bar{x} \in X_E$, i.e. an efficient point of (P). We assume that $f(\bar{x}) \notin T$. There are two possibilities: First, $f(\bar{x})$ lies above T, i.e. $f(\bar{x}) \in \hat{Z} \setminus (T - \mathbb{R}^m_+)$. By $f(\bar{x}) \notin T - \mathbb{R}^m_+$ it holds for all $p \in \mathcal{L}_{LUB} : f(\bar{x}) \not\leq p$. In particular $f(\bar{x}) \notin S$, otherwise we have a contradiction to Definition 3.2 (i). Using the definition of S, see (3), it follows that there exists a point $q \in \mathcal{L}_{PNS}$ with $q \leq f(\bar{x})$. Since q is an image of a feasible point and $\bar{x}$ is efficient $q = f(\bar{x})$ holds. But $f(\bar{x}) \notin T$ which contradicts $q \in \mathcal{L}_{PNS} \subseteq T$. The other case is $f(\bar{x}) \in (T - \mathbb{R}^m_+) \setminus T$. Hence there is a local upper bound $\bar{p} \in \mathcal{L}_{LUB}$ with $f(\bar{x}) < \bar{p} - \frac{\varepsilon}{2}e$, which leads to the chain of inequalities: $f_\alpha(\bar{x}) \leq f(\bar{x}) < \bar{p} - \frac{\varepsilon}{2}e$. By Lemma 4.4 there is some box $X \in \mathcal{L}_{S,1}$ with $\bar{x} \in X$ and thus $\bar{p} - \frac{\varepsilon}{2} \in f_\alpha(X) + \mathbb{R}^m_+$. This is a contradiction to (6). Therefore $f(\bar{x}) \in T$. □

Remark 4.9. *As a consequence of Theorem 4.8 we also have $f(X^0) \subseteq T + \mathbb{R}^m_+$, i.e. $f(X^0) \cap (\mathbb{R}^m \setminus (T + \mathbb{R}^m_+)) = \emptyset$, which means that no image of a feasible point of (P) lies below T. This is due to the fact that the ordering cone $\mathbb{R}^m_+$ is a pointed closed convex cone and $f(X^0)$ is a compact set, and thus external stability holds, compare with [38, Theorem 3.2.9].*

Next we want to show the (ε, δ)-minimality of the output set $\mathcal{A}$. Recall that the set $\mathcal{X}$ is calculated individually for every considered subbox X^* in Algorithm 3, line 8. All $x \in \mathcal{X}$ can be possible ε-minimal points of (P). In Algorithm 4 line 23 two conditions are checked and we will prove in the next lemmas that for those $x \in \mathcal{X}$ ε-minimality indeed holds.

Lemma 4.10. *Every subbox $X \in \mathbb{R}^n$ of X^0, which is not discarded in the third `while`-loop of Algorithm 4 and with $\omega(X) < \sqrt{\frac{\varepsilon}{\alpha}}$ contains a point $\tilde{x}$ which is ε-minimal.*

Proof. Set $\tilde{\delta} := \sqrt{\frac{\varepsilon}{\alpha}}$ and let $X \subseteq X^0$ be a box with $\omega(X) < \tilde{\delta}$. With Remark 2.2 we know that for arbitrary $x \in X$ and all $j = 1, \ldots, m$ holds: $f_j(x) - f_{j,\alpha}(x) = |f_j(x) - f_{j,\alpha}(x)| \leq \frac{\alpha}{2}\omega(X)^2 < \frac{\alpha}{2} \cdot \frac{\varepsilon}{\alpha} = \frac{\varepsilon}{2}$. By f_α being a convex underestimator of f (componentwise) we obtain

$$f_{j,\alpha}(x) \leq f_j(x) < f_{j,\alpha}(x) + \frac{\varepsilon}{2} \text{ for every } j \in \{1, \ldots, m\} \text{ and all } x \in X, \tag{12}$$

which is equivalent to $f(x) \in \left(\left\{f_\alpha(x) + \frac{\varepsilon}{2}e\right\} - \operatorname{int}(\mathbb{R}^m_+)\right) \cap \left(\{f_\alpha(x)\} + \mathbb{R}^m_+\right)$. The box X was not discarded in the third `while`-loop. Hence Algorithm 3 applied to X has to obtain the output $\mathcal{D} = 0$ and a list $\mathcal{X} \neq \emptyset$, see line 8 of Algorithm 3. Now consider an $\tilde{x} \in X$, which was calculated in line 6 of Algorithm 3 w.r.t. the corresponding local upper bound $p^s \in \mathcal{L}_{LUB}$. Note that for the solution $(\tilde{x}, \tilde{t})$ of $(P_{p^s,X})$ we have $-\frac{\varepsilon}{2} \leq \tilde{t} \leq 0$, for $-\frac{\varepsilon}{2} \leq \tilde{t}$ see Lemma 4.6. Suppose now there is an $\hat{x} \in X^0$ with $f(\hat{x}) \leq f(\tilde{x}) - \varepsilon e$ and $f(\hat{x}) \neq f(\tilde{x}) - \varepsilon e$, which is equivalent to $f(\hat{x}) \in \{f(\tilde{x})\} - \{\varepsilon e\} - (\mathbb{R}^m_+ \setminus \{0_m\})$. Hence with the above shown property (12) we conclude

$$\begin{aligned} f(\hat{x}) &\in \{f(\tilde{x})\} - \{\varepsilon e\} - (\mathbb{R}^m_+ \setminus \{0_m\}) \\ &\subseteq \left(\left(\left\{f_\alpha(\tilde{x}) + \frac{\varepsilon}{2}e\right\} - \operatorname{int}(\mathbb{R}^m_+)\right) \cap \left(\{f_\alpha(\tilde{x})\} + \mathbb{R}^m_+\right)\right) - \{\varepsilon e\} - (\mathbb{R}^m_+ \setminus \{0_m\}) \\ &\subseteq \left\{f_\alpha(\tilde{x}) - \frac{\varepsilon}{2}e\right\} - \operatorname{int}(\mathbb{R}^m_+), \end{aligned} \tag{13}$$

which is equivalent to $f(\hat{x}) < f_\alpha(\tilde{x}) - \frac{\varepsilon}{2}$.

Because of $(\tilde{x}, \tilde{t})$ being a feasible point of $(P_{p^s,X})$ we have $p^s + \tilde{t}e \geq f_\alpha(\tilde{x})$. Since this and $\tilde{t} \leq 0$ we obtain $f_\alpha(\tilde{x}) \leq p^s$. With (13) it follows $f(\hat{x}) < p^s - \frac{\varepsilon}{2}$ and thus $f(\hat{x})$ is not in $T + \mathbb{R}^m_+$, which contradicts Remark 4.9 that no feasible image is below T. Hence $\tilde{x}$ is ε-minimal. □

Lemma 4.11. *Let $\mathcal{A}$ be the set generated by Algorithm 4. Then $\tilde{x} \in \mathcal{A}$ is an ε-minimal point of (P).*

Proof. Let $\tilde{x}$ be an arbitrary element of $\mathcal{A}$. If $\tilde{x}$ is added in line 29 to $\mathcal{A}$, the point $f(\tilde{x})$ is an element of $\mathcal{L}_{PNS}$ and thus by Lemma 3.4 there is a local upper bound $\bar{p} \in \mathcal{L}_{LUB}$ with $f(\tilde{x}) \leq \bar{p}$. Now suppose there is an $\hat{x} \in X^0$ with $f(\hat{x}) \leq f(\tilde{x}) - \varepsilon e$ and $f(\hat{x}) \neq f(\tilde{x}) - \varepsilon e$. With $f(\tilde{x}) \leq \tilde{p}$ we obtain $f(\hat{x}) \leq \tilde{p} - \varepsilon e$ and hence $f(\hat{x}) \notin T$, but below T. But that contradicts Remark 4.9 that no feasible image is below T.

If $\tilde{x}$ is added in line 24, $\tilde{x}$ belongs to a box X^* with $\omega(X^*) \leq \delta$ which was not discarded in the third `while`-loop and thereby $\tilde{x} \in \mathcal{X}$. Hence there is a local upper bound $\bar{p} \in \mathcal{L}_{LUB}$ with $(\tilde{x}, \tilde{t})$ is a solution of $(P_{\bar{p},X^*})$ and $-\frac{\varepsilon}{2} \leq \tilde{t} \leq 0$, for $-\frac{\varepsilon}{2} \leq \tilde{t}$ see Lemma 4.6. As the first condition in line 23 of Algorithm 4 is satisfied we have $f(\tilde{x}) \leq \tilde{p}$ for a local upper bound $\tilde{p} \in \mathcal{L}_{LUB}$. With the same argumentation as at the beginning of this proof we can show that $\tilde{x}$ is ε-minimal.

If the first condition in line 23 is not satisfied, but the second one, i.e $\omega(X^*) < \sqrt{\frac{\varepsilon}{\alpha}}$, we have to show that each $x \in \mathcal{X}$ is ε-minimal. For this we refer to the proof of Lemma 4.10, because the points in $\mathcal{X}$ of X^* with $\omega(X^*) < \sqrt{\frac{\varepsilon}{\alpha}}$ are exactly those points, which are the ε-minimal points from Lemma 4.10. □

Theorem 4.12. *Let $\mathcal{A}$ be the set generated by Algorithm 4. Then $\mathcal{A}$ is an (ε, δ)-minimal set of (P).*

Proof. With Lemma 4.11 we know that $\mathcal{A}$ contains only ε-minimal points.

Now let $y \in X_E$ be an efficient solution of (P). With Lemma 4.4 we know that a box Y containing y can not be discarded in any `while`-loop. Thus, choose now a box $Y \in \mathcal{L}_{S,3}$ with $y \in Y$. This box exists because the algorithm terminates, see Lemmas 4.1 to 4.3. Because of the procedure before storing Y to $\mathcal{L}_{S,3}$ in line 24 of Algorithm 4 there will be an $x \in \mathcal{X} \subseteq Y$ added to the set $\mathcal{A}$. Moreover $\omega(Y) \leq \delta$ and thus $\|x - y\| \leq \delta$. □

5 Handling of constraints

In the following we explain which difficulties arise, if we are considering MOPs like (MOP) with convex inequalities described by $g_r \colon \mathbb{R}^n \to \mathbb{R},\ r = 1, \ldots, p$. The first issue is the handling of the list $\mathcal{L}_{PNS}$. As we explained after Algorithm 2, every point of $\mathcal{L}_{PNS}$ has to be an image of a feasible point of (MOP). This feasibility is already ensured by choosing the images of a solution of the minimization of the convex underestimator over the feasible set $M^* := \{x \in X^* \mid g(x) \leq 0_p\}$.

The new discarding test can be easily applied to convex constraints by replacing X^0 by M, which is the feasible set given by convex constraints, and X^* by M^* in every optimization problem which has to be solved. In particular that is the case when the ideal point of a convex underestimator is determined and for solving $(P_{\bar{p},X^*})$. However, this may result in an infeasible set M^*. In case the feasible set is empty we can discard the box. Numerically it might be difficult to verify that the feasible set is indeed empty. A fast and efficient sufficient condition was proposed in [14] and uses interval arithmetic to obtain lower and upper bounds of $g_r,\ r = 1, \ldots, p$. For dealing with nonconvex constraints and finding feasible points for such optimization problems we refer to [24]. Furthermore in [10] an approach is presented which uses the solution of the dual problem to identify infeasible boxes.

6 Numerical Results

In this section we illustrate our algorithm on some test instances from the literature. We also solve an application problem which arises in engineering in the context of Lorentz force velocimetry.

6.1 Test instances

First we compare the cases (II) and (III) from the beginning of Section 3.1. Recall that (II) uses the ideal point of the convex underestimators to obtain lower bounds and compares them with $\mathcal{L}_{PNS}$ only. Case (III) was the idea of the new discarding test. For a better comparability we restrict our algorithm to the first `while`-loop with a termination rule $\omega(X^l) < \delta$.

Test instance 6.1. *This test instance is based on [16] and is scalable in $n \in \mathbb{N}$.*

$$f(x) = \begin{pmatrix} 1 - \exp\left(-\sum_{i=1}^{n}\left(x_i - \frac{1}{\sqrt{n}}\right)^2\right) \\ 1 - \exp\left(-\sum_{i=1}^{n}\left(x_i + \frac{1}{\sqrt{n}}\right)^2\right) \end{pmatrix} \quad \text{with } X^0 = \left[\begin{pmatrix} -2 \\ \vdots \\ -2 \end{pmatrix}, \begin{pmatrix} 2 \\ \vdots \\ 2 \end{pmatrix}\right] \in \mathbb{R}^n.$$

For the number of iterations, the calculation time t and the number of boxes in the solution list $\mathcal{L}_S$, we obtain the results presented in Table 1.

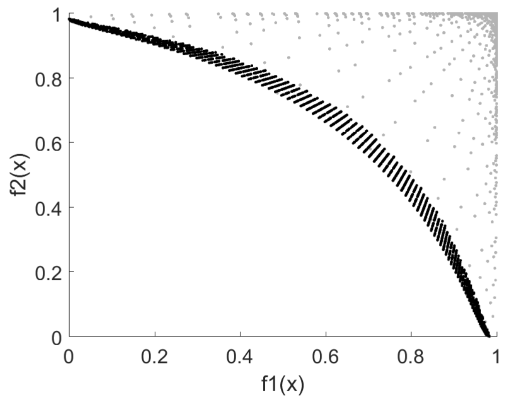
	case (II) only ideal point			case (III) new discarding test		
n	# iterations	t [s]	$\lvert\mathcal{L}_S\rvert$	# iterations	t [s]	$\lvert\mathcal{L}_S\rvert$
1	41	3.5970	34	41	4.1928	34
2	456	39.4317	262	359	38.1401	210
3	6283	626.7255	3434	3055	364.1089	1268
4	78965	1.0014e+04	42540	20966	2.9587e+03	7644

Table 1: Results for Test instance 6.1 with $\delta = 0.1$.

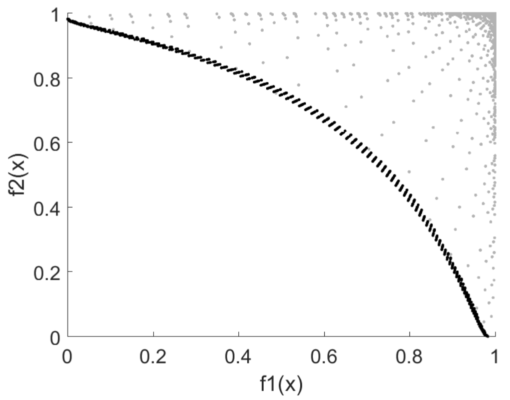
The pictures in Figure 7 show the results in the image space for $n = 3$. The image set is represented by some image points in grey which are obtained by a discretization of the feasible set X^0. The black points are the images of the midpoints of the boxes of the list $\mathcal{L}_S$.

It can be seen that we can decrease the number of iterations with the new procedure, which is clear, because calculating the ideal point by minimizing convex underestimators is also a part of our developed discarding test (III). Additionally for $n \geq 2$ the new procedure is faster while we have to solve more optimization problems on each subbox. In the case $n = 3$ the approximation of the nondominated set is much tighter as can be seen from the Figure 7. In the other cases for n we obtained similar results.

For illustrating the whole procedure with all three `while`-loops we choose $\varepsilon = 0.05$ and $\delta = 0.1$. The plots in Figures 8a and 8b are showing the partitioning of the feasible set after the second and third `while`-loop. Medium grey boxes are those which have been discarded in the first `while`-loop. The light grey boxes were discarded after the second `while`-loop. The dark grey boxes were not discarded after the second and third `while`-loop respectively. In Figure 8b there are additionally black points which are the points from



(a) Image space with discarding test based on interval arithmetic.



(b) Image space with new discarding test.

Figure 7: Test instance 6.1 with $n = 3, \delta = 0.1$

the approximation set $\mathcal{A}$. Furthermore the new light grey boxes compared to Figure 8a were discarded within the third `while`-loop. Figure 8c shows the image space of the test function. The black points are the images of the approximation set $\mathcal{A}$. Additionally in Figure 8d the obtained $\frac{\varepsilon}{2}$-tube is shown.

The results of the next test instance show that our new algorithm is also able to find global solutions in case there are also only locally optimal solutions.

Test instance 6.2. *This test instance was proposed in [8]:*

$$f(x) = \begin{pmatrix} x_1 \\ \frac{1}{x_1}\left(2 - \exp(-(\frac{x_2-0.4}{0.004})^2) - 0.8\exp(-(\frac{x_2-0.6}{0.4})^2)\right) \end{pmatrix},$$

with $X^0 = \left[\begin{pmatrix}0.1\\0\end{pmatrix}, \begin{pmatrix}1\\1\end{pmatrix}\right] \in \mathbb{R}^2$. *The globally efficient points are* $(\tilde{x}_1, \tilde{x}_2)$ *with* $\tilde{x}_2 \approx 0.2$ *and* $\tilde{x}_1 \in [0.1, 1]$. *But this test instance has also locally efficient points with* $\tilde{x}_2 \approx 0.6$ *and* $\tilde{x}_1 \in [0.1, 1]$.

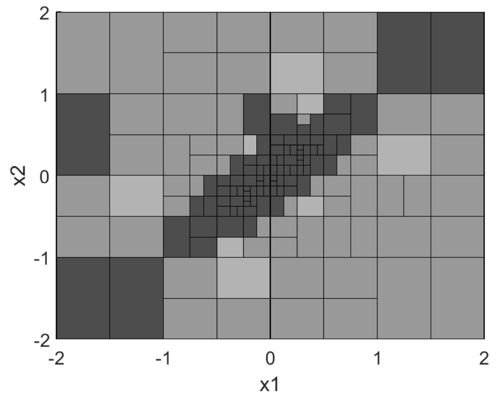
Figure 9 shows the results of our algorithm on this test instance. Obviously the global solutions can be found next to some few ε-minimal points with $x_1 \approx 0.1$. For example a weighted sum approach with a local optimization solver as SQP may be able to find the local solutions only.

The next test instance has three objective functions. Moreover we added a convex constraint g.

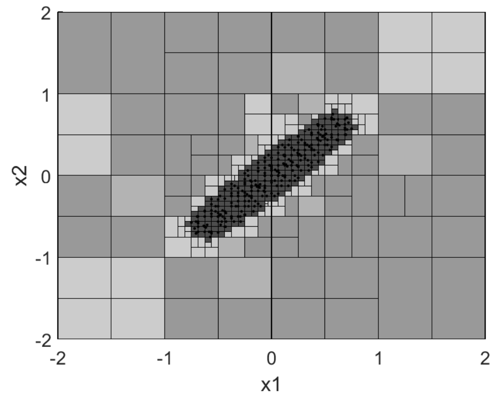
Test instance 6.3. *This test instance without the convex constraint was introduced in [42]:*

$$f(x) = \begin{pmatrix} 0.5(x_1^2 + x_2^2)^2 + \sin(x_1^1 + x_2^2) \\ \frac{(3x_1-2x_2+4)^2}{8} + \frac{(x_1-x_2+1)^2}{27} + 15 \\ \frac{1}{x_1^2-x_2^2-1} - 1.1\exp(-x_1^2 - x_2^2) \end{pmatrix} \text{ with } X^0 = \left[\begin{pmatrix}-3\\-3\end{pmatrix}, \begin{pmatrix}3\\3\end{pmatrix}\right] \in \mathbb{R}^2$$

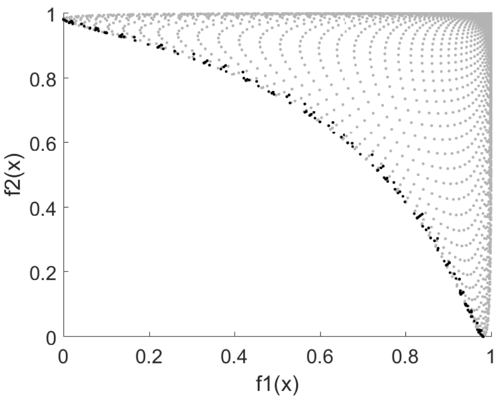
and in addition $g(x) = x_1^2 + x_2^2 - 4$.



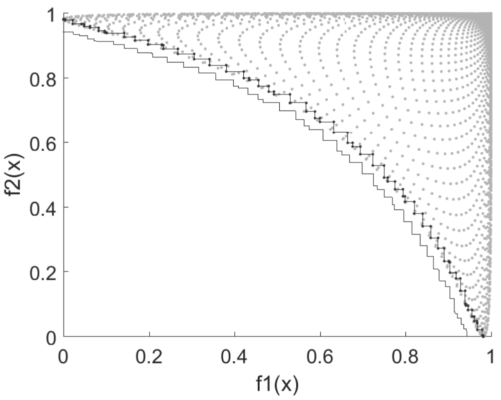
(a) Partition of the feasible set after second `while`-loop, 66 discarded boxes.



(b) Partition of the feasible set after third `while`-loop, 151 discarded boxes, 279 points in $\mathcal{A}$.

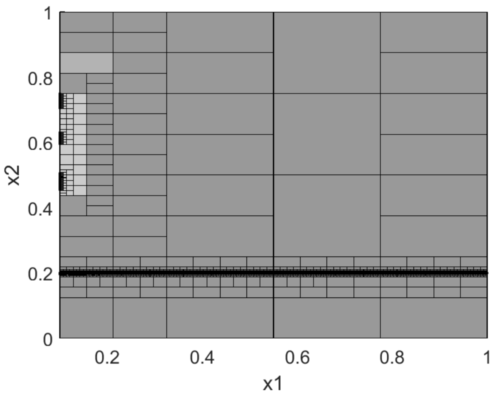


(c) Image space and images of the (ε, δ)-minimal set $\mathcal{A}$.

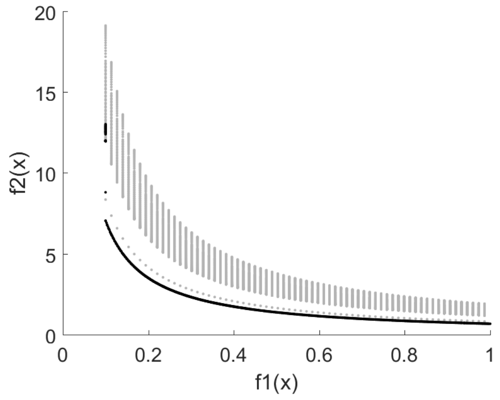


(d) Image space, images of the (ε, δ)-minimal set A and $\frac{\varepsilon}{2}$-tube.

Figure 8: Test instance 6.1 with $n = 2, \varepsilon = 0.05$ and $\delta = 0.1$



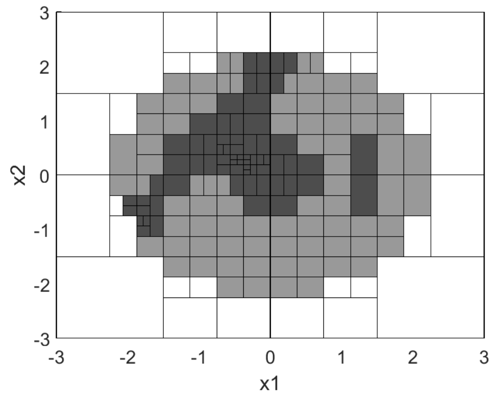
(a) Partition of the feasible set after third `while`-loop, 2039 discarded boxes, 648 points in $\mathcal{A}$.



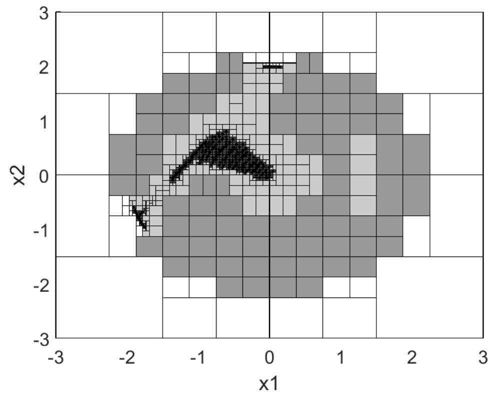
(b) Image space and images of the (ε, δ)-minimal set $\mathcal{A}$.

Figure 9: Test instance 6.2 with $\varepsilon = 0.01$ and $\delta = 0.01$

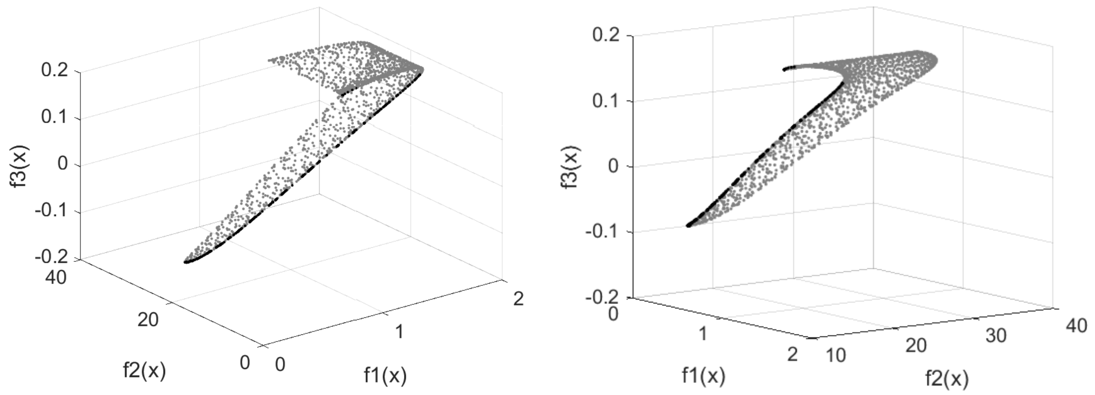
Figure 10 shows the results of this test instance. Additionally to the meanings of grey shades in the former graphical results the white boxes are boxes, which were discarded because of infeasibility. In this example no box was discarded in the second `while`-loop. Moreover we can see in Figures 10b and 10c that our algorithm is also able to find non-connected areas of efficient or nondominated points.



(a) Partition of the feasible set after second `while`-loop, 108 discarded boxes.



(b) Partition of the feasible set after third `while`-loop, 315 discarded boxes, 360 points in $\mathcal{A}$.



(c) Image space and images of the (ε, δ)-minimal set $\mathcal{A}$ in different perspectives.

Figure 10: Test instance 6.3 with $\varepsilon = 0.1$ and $\delta = 0.1$

6.2 Application in Lorentz force velocimetry

We have also applied our algorithm to a problem which arises in the optimization of a measurement technique known as Lorentz force velocimetry (LFV). In LFV the aim is to measure for example the velocity of fluids. The technique is based on measurements of the Lorentz force that occurs due to the flow of a conductive fluid under the influence of a variable magnetic field. For generating a measurement system in our problem setting n dipoles have to be arranged around a cylinder with an electric conductive fluid. We assume that the dipoles are placed at equidistant positions $r^i = (0, \cos\gamma_i, \sin\gamma_i)^T,\ i = 1, \ldots, n$ around the cylinder and aim on finding the optimal magnetic orientation. We assume that all dipole moments have the same magnitude m. The orientation of an individual dipole i is then represented in terms of polar and azimuthal angles θ_i and φ_i, where φ_i is splitted into the two angles β_i and the fixed γ_i. Thus the magnetic moment vector m^i is $m^i(\theta_i, \beta_i) = (m \cos\theta_i, m \sin\theta_i \cos(\beta_i + \gamma_i), m \sin\theta_i \sin(\beta_i + \gamma_i))$.

The first objective is to maximize the absolute value of the axial force component as in [4]. Since the dipoles are on a circle of radius H in the plane $x = 0$ the force can be analytically expressed. The self-interaction term for a dipole i and the mutual interaction term between dipoles i and j are

$$\begin{aligned} F_s(\beta_i, \theta_i) &= \frac{45\pi}{4096} \cdot \frac{v\sigma R^4 \mu_0^2 m^2}{128\pi H^7}[355 + 25\cos(2\theta_i) + 266\cos(2\beta_i)\sin^2\theta_i] \text{ and} \\ F_m(\beta_i, \beta_j, \theta_i, \theta_j) &= \tfrac{45\pi}{1024} \cdot \tfrac{v\sigma R^4 \mu_0^2 m^2}{128\pi H^7}[10(5 + 14\cos(\gamma_i - \gamma_j))\cos\theta_i \cos\theta_j \\ &\quad + \sin\theta_i \sin\theta_j(49\cos(\gamma_i - \gamma_j - \beta_i - \beta_j) + 35\cos(\beta_i - \beta_j) \\ &\quad +25\cos(\gamma_i - \gamma_j + \beta_i - \beta_j) + 105\cos(\gamma_i - \gamma_j - \beta_i + \beta_j) \\ &\quad +35\cos(\beta_i + \beta_j) + 49\cos(\gamma_i - \gamma_j + \beta_i + \beta_j))] \end{aligned}$$

respectively. For the resulting interaction force we obtain

$$F_x(\beta, \theta) = \sum_{i=1}^{n} F_s(\beta_i, \theta_i) + \sum_{i<j} F_m(\beta_i, \beta_j, \theta_i, \theta_j).$$

The constants are the velocity of the fluid v, the electric conductivity σ, the radius of the cylinder R and the vacuum permeability $\mu_0 = 4\pi \cdot 10^{-7}$Vs/Am.

The second objective is a minimization of the interaction potential energy between the dipoles. The reason for that is that arrangements with a high interaction potential energy are more difficult to realize. The vector $r_{i,j}$ represents the vector between the positions of both dipoles: $r_{i,j} = r^j - r^i$. The function is the sum of all energies between every pair of dipoles:

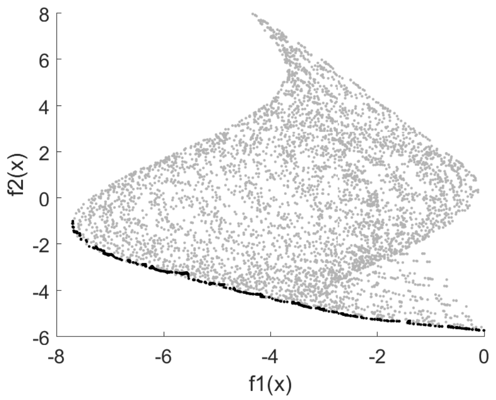
$$V(\beta, \theta) = \sum_{i<j} \frac{\mu_0}{4\pi |r_{i,j}|^5}[|r_{i,j}|^2 m_i(\theta_i, \beta_i) m_j(\theta_j, \beta_j) - 3(m_i(\theta_i, \beta_i) r_{i,j})(m_j(\theta_j, \beta_j) r_{i,j})].$$

To reduce the dimension of the decision space we fix $\theta_i = \frac{\pi}{2}$ for all $i = 1, \dots n$. Thus and because of symmetry of the arrangements the feasible set of interesting angles β is given by

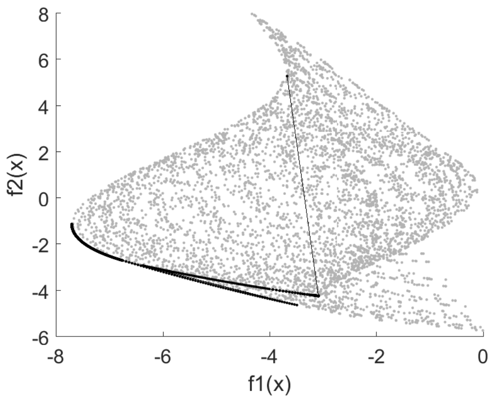
$$X^0 = \left[(-\frac{\pi}{2}, -\pi, \dots, -\pi)^T, (\frac{\pi}{2}, \pi, \dots, \pi)^T\right] \in \mathbb{R}^n.$$

All constant coefficients are set to 1 and we scale the objective functions with -0.1 (to switch from maximization to minimization) and 100, respectively, to obtain different accuracies for both objective functions.

The following plot in Figure 11a shows the image space after executing the algorithm with $\varepsilon = \delta = 0.5$ for three dipoles. Figure 11b is one result after using a weighted sum approach with a standard SQP solver for the scalarized problems. The black points connected by thin black lines are the solutions of minimizing a weighted sum of the objectives and are also used as the next starting point for the next weighted sum. Obviously this approach was able to find local solutions first and only after some iterations some of the global solutions. Moreover the lower right part of the nondominated set was not found. We tried different starting points, but never obtained an approximation of the whole nondominated set.



(a) Image space after Algorithm 4.



(b) Image space after a weighted sum approach with a random starting point.

Figure 11: Graphical results for 3 dipoles on a circle around a cylinder.

7 Conclusions

We have combined the ideas of convex underestimators with techniques from convex multiobjective optimization and the idea of local upper bounds from general multiobjective optimization to obtain an efficient discarding test. Numerical experiments show that we get good results even while we have chosen large values for ε and δ. That is due to the fact that our estimations in some proofs are quite rough. The current implementation does not use other, more simple, discarding tests as already proposed in the literature. By using also those tests further time savings are expected. An example for that are monotonicity tests as proposed in [14]. In [44] also interesting bounds based on Lipschitz constants are derived and it would be of interest to explore possible combinations. Also the handling of nonconvex constraints should be considered further. But the same difficulties as for scalar valued global optimization algorithms will arise. These difficulties are for instance discussed in [24].

8 Acknowledgments

The authors wish to thank Prof. Dr. A. Löhne and Dr. B. Weißing (Friedrich-Schiller-Universität Jena) as well as Prof. Dr. K. Klamroth (Bergische Universität Wuppertal) for valuable discussions. Moreover we thank PD Dr. T. Boeck (TU Ilmenau) for the cooperation concerning Lorentz force velocimetry and B.Sc. S. Rocktäschel (TU Ilmenau) for supporting the implementations. Furthermore the first author thanks Thuringian State Graduate Support Regulation, Carl-Zeiss-Stiftung and DFG-funded Research Training Group 1567 "Lorentz Force Velocimetry and Lorentz Force Eddy Current Testing" for financial support.

References

[1] C.S. Adjiman, S. Dallwig, C.A. Floudas, and A. Neumaier. A Global Optimization Method, αBB, for General Twice-Differentiable Constrained NLPs: I - Theoretical

Advances. *Computers & Chemical Engineering*, 22(9):1137–1158, 1998.

[2] Ch. Audet, P. Hansen, B. Jaumard, and G. Savard. A branch and cut algorithm for nonconvex quadratically constrained quadratic programming. *Mathematical Programming. Series A. Series B*, 87(1 (A)):131–152, 2000.

[3] H.P. Benson. An outer approximation algorithm for generating all efficient extreme points in the outcome set of a multiple objective linear programming problem. *Journal of Global Optimization*, 13(1):1–24, 1998.

[4] T. Boeck, D. Terzijska, and G. Eichfelder. Maximum electromagnetic drag configurations for a translating conducting cylinder with distant magnetic dipoles. *Journal of Engineering Mathematics*, 2017. https://doi.org/10.1007/s10665-017-9916-8.

[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] E.F. Campana, M. Diez, G. Liuzzi, S. Lucidi, R. Pellegrini, V. Piccialli, F. Rinaldi, and A. Serani. A Multi-objective DIRECT algorithm for ship hull optimization. *Computational Optimization and Applications*, 2017. https://doi.org/10.1007/s10589-017-9955-0.

[7] A.L. Custódio and J.F.A. Madeira. MultiGLODS: Global and Local Multiobjective Optimization Using Direct Search. submitted for publication, 2017.

[8] K. Deb. Multi-objective Genetic Algorithms: Problem Difficulties and Construction of Test Problems. *Evolutionary Computation*, 7(3):205–230, 1999.

[9] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transaction on Evolutionary Computation*, 6(2):181–197, 2002.

[10] M. Dür. Dual bounding procedures lead to convergent BranchandBound algorithms. *Mathematical Programming*, 91(1):117–125, 2001.

[11] M. Ehrgott. *Multicriteria Optimization*. Springer, 2005.

[12] M. Ehrgott, L. Shao, and A. Schöbel. An approximation algorithm for convex multi-objective programming problems. *Journal of Global Optimization*, 50(3):397–416, 2011.

[13] G. Eichfelder. *Adaptive Scalarization Methods in Multiobjective Optimization*. Springer, 2008.

[14] J. Fernández and B. Tóth. Obtaining the efficient set of nonlinear biobjective optimization problems via interval branch-and-bound methods. *Computational Optimization and Applications*, 42(3):393–419, 2009.

[15] J. Fliege and A.I. F. Vaz. A method for constrained multiobjective optimization based on SQP techniques. *SIAM Journal on Optimization*, 26(4):2091–2119, 2016.

[16] C. Fonseca and P. Fleming. Multiobjective genetic algorithms made easy: selection sharing and mating restriction. In *Proceedings 1st International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications*, pages 45–52, 1995.

[17] D. Gourion and D.T. Luc. Finding efficient solutions by free disposal outer approximation. *SIAM Journal on Optimization*, 20(6):2939–2958, 2010.

[18] C. Gutiérrez, B. Jiménez, and V. Novo. A Unified Approach and Optimality Conditions for Approximate Solutions of Vector Optimization Problems. *SIAM Journal on Optimization*, 17(3):688–710, 2006.

[19] W. W. Hager and D. T. Phan. An ellipsoidal branch and bound algorithm for global optimization. *SIAM Journal on Optimization*, 20(2):740–758, 2009.

[20] E.M.T. Hendrix and B.G. Tóth. *Introduction to Nonlinear and Global Optimization*. Springer, 2010.

[21] J. Jahn. Multiobjective search algorithm with subdivision technique. *Computational Optimization and Applications*, 35(2):161–175, 2006.

[22] J. Jahn. *Vector Optimization - Theory, Applications, and Extensions*. Springer, 2011.

[23] D.R. Jones, C.D. Perttunen, and B.E. Stuckman. Lipschitzian optimization without the Lipschitz constant. *Journal of Optimization Theory and Applications*, 79(1):157–181, 1993.

[24] P. Kirst, O. Stein, and P. Steuermann. Deterministic upper bounds for spatial branch-and-bound methods in global minimization with nonconvex constraints. *TOP*, 23(2):591–616, 2015.

[25] K. Klamroth. private communication, 2017.

[26] K. Klamroth, R. Lacour, and D. Vanderpooten. On the representation of the search region in multi-objective optimization. *European Journal of Operational Research*, 245(3):767–778, 2015.

[27] S.S. Kutateladze. Convex ε-programming. *Soviet mathematics - Doklady*, 20:391–393, 1979.

[28] D. Lera and Y.D. Sergeyev. Acceleration of univariate global optimization algorithms working with Lipschitz functions and Lipschitz first derivatives. *SIAM Journal on Optimization*, 23(1):508–529, 2013.

[29] M. Locatelli and F. Schoen. *Global Optimization: Theory, Algorithms, and Applications*. SIAM, 2013.

[30] A. Löhne, B. Rudloff, and F. Ulus. Primal and dual approximation algorithms for convex vector optimization problems. *Journal of Global Optimization*, 60(4):713–736, 2014.

[31] P. Loridan. ε-solutions in vector minimization problems. *Journal of Optimization Theory and Applications*, 43(2):265–276, 1984.

[32] A. Lovison. Singular continuation: generating piecewise linear approximations to Pareto sets via global analysis. *SIAM Journal on Optimization*, 21(2):463–490, 2011.

[33] C.D. Maranas and C. A. Floudas. Global minimum potential energy conformations of small molecules. *Journal of Global Optimization*, 4(2):135–170, 1994.

[34] D. Mueller-Gritschneder, H. Graeb, and U. Schlichtmann. A successive approach to compute the bounded Pareto front of practical multiobjective optimization problems. *SIAM Journal on Optimization*, 20(2):915–934, 2009.

[35] A. Neumaier. *Interval methods for systems of equations.* Cambridge University Press, 1990.

[36] P.M. Pardalos, A. Žilinskas, and J. Žilinskas. *Non-Convex Multi-Objective Optimization.* Springer, 2017.

[37] S.M. Rump. INTLAB - INTerval LABoratory. In Tibor Csendes, editor, *Developments in Reliable Computing*, pages 77–104. Kluwer Academic Publishers, 1999.

[38] Y. Sawaragi, H. Nakayama, and T. Tanino. *Theory of Multiobjective Optimization.* Academic Press, 1985.

[39] D. Scholz. *Modern Electrodynamics.* Springer, 2012.

[40] M. Schulze Darup and M. Mönnigmann. Improved Automatic Computation of Hessian Matrix Spectral Bounds. *SIAM Journal on Scientific Computing*, 38(4):A2068–A2090, 2016.

[41] H. Tuy and R. Horst. Convergence and restart in branch-and-bound algorithms for global optimization. Application to concave minimization and d.c. optimization problems. *Mathematical Programming. Series A. Series B*, 41(2 (A)):161–183, 1988.

[42] R. Viennet, C. Fonteix, and I. Marc. Multicriteria optimization using a genetic algorithm for determining a Pareto set. *International Journal of Systems Science*, 27(2):255–260, 1996.

[43] L. Wang, H. Ishida, T. Hiroyasu, and M. Miki. Examination of multi-objective optimization method for global search using DIRECT and GA. *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, pages 2446–2451, 2008.

[44] A. Žilinskas and J. Žilinskas. Adaptation of a one-step worst-case optimal univariate algorithm of bi-objective Lipschitz optimization to multidimensional problems. *Commun Nonlinear Sci Numer Simulat*, 21(1-3):89–98, 2016.