Pure and Applied Functional Analysis
Volume 3, Number 4, 2018, 603–623

Yokohama Publishers
ISSN 2189-3764 ONLINE JOURNAL
© Copyright 2018

A DERIVATIVE-FREE ROOTED TREE METHOD IN NONCONVEX SET OPTIMIZATION

JOHANNES JAHN

Abstract. This paper presents a numerical method for the solution of nonconvex set optimization problems with the minmax less order relation, which has advantages in practice in contrast to other order relations but which is more difficult to handle. Because of the high numerical effort this algorithm is implemented on a CPU and GPU. This method further extends a pattern search method [9] to this complex class of set optimization problems. The iteration sequence generates a rooted tree. The results include a complete convergence theory and numerical investigations for problems with sets up to dimension four.

1. Introduction

Problems of set optimization are optimization problems with a set-valued objective map. For example, such problems occur, if there are uncertainties in a real-valued objective function so that function values may vary in a specified region (compare [8, Example 14.3]). In practice, these set optimization problems arise in socio-economics [14], the design of photovoltaic power plants [3] and general energy systems [2], among others.

In this paper we investigate the numerical solution of set optimization problems equipped with the minmax order relation introduced in [10, Def. 3.5]. This special order relation is more helpful in real-world applications than other relations known in the literature. Since one has to know the subsets of all minimal and maximal elements of a given set, the minmax order relation is even harder to treat from a numerical point of view. In order to compensate this higher numerical effort the computation of expensive parts is carried out in parallel. Although there are various possibilities for parallelization, we restrict ourselves to vector computations on Nvidia's graphics processing units (GPUs) because there are already good experiences for the computation of minimal and maximal elements of a set ([13]). GPUs can handle sets with parameterized boundary in a very effective way.

In addition to the afore-mentioned computational aspects one also needs a theoretical measure to formulate an appropriate step size rule which ensures the convergence of descent methods. The necessary theory is developed in the setting of real normed spaces; for the algorithm and the convergence investigations a restriction to the space $\mathbb{R}^n$ is not necessary.

2010 *Mathematics Subject Classification.* 90C56, 06A75, 65Y05.
Key words and phrases. set optimization, minmax less order relation, GPU computing.

In the case of a real-valued objective function Conn, Scheinberg and Vicente [5] highlight the usefulness of derivative-free methods in nonlinear programming. For set-valued objective maps and standard order relations pattern search methods as special descent methods have shown to be appropriate for the numerical solution of convex set optimization problems ([9]). In this paper pattern search methods are extended to nonconvex set optimization problems and they are not restricted to a single iteration path but they are extended to a whole tree consisting of iteration pathes starting at the same point. In this sense a rooted tree method is presented.

This paper is organized as follows. Section 2 presents basic results for the minmax order relation together with a measure being useful for the algorithmic treatment of this special order relation. In Section 3 a derivative-free rooted tree descent method is given and convergence of this algorithm is investigated. Numerical results are given for problems with sets in $\mathbb{R}^2$, $\mathbb{R}^3$ and even $\mathbb{R}^4$ in Section 4. A short appendix shows some algorithmic parts carried out on Nvidia's GPUs with CUDA C and Matlab.

2. Basic results

Throughout this paper we investigate set optimization problems in the following setting.

Assumption 2.1. Let S be a nonempty subset of a real linear space X, let C be a convex cone in a real linear space Y, and let $F : S \rightrightarrows Y$ be a set-valued map with $F(x) \neq \emptyset$ for all $x \in S$.

Under Assumption 2.1 we now investigate the set optimization problem

$$\min_{x \in S} F(x). \tag{2.1}$$

Minimal solutions of problem (2.1) are defined using the minmax less order relation.

Definition 2.2. Let Assumption 2.1 be satisfied.

(a) The set of all *minimal* elements of a nonempty subset A of Y is denoted by

$$\min A := \{a \in A \mid (\{a\} - C) \cap A \subset \{a\} + C\},$$

and

$$\max A := \{a \in A \mid (\{a\} + C) \cap A \subset \{a\} - C\}$$

denotes the set of all *maximal* elements of the set A.

(b) Let A, B be subsets of Y with $\min A, \min B, \max A, \max B \neq \emptyset$. Then the *set less* order relation $\preccurlyeq_s$ is defined by

$$A \preccurlyeq_s B \quad :\Longleftrightarrow \quad B \subset A + C \text{ and } A \subset B - C.$$

The order relation $\preccurlyeq$ defined by

$$A \preccurlyeq B \quad :\Longleftrightarrow \quad \min A \preccurlyeq_s \min B \text{ and } \max A \preccurlyeq_s \max B$$

is called *minmax less order relation.*

(c) $\bar{x} \in S$ is called a *minimal solution* of the set optimization problem (2.1) iff $F(\bar{x})$ is a minimal set of the system of sets $F(x)$ (with arbitrary $x \in S$), i.e.

$$F(x) \preccurlyeq F(\bar{x}),\ x \in S \implies F(\bar{x}) \preccurlyeq F(x).$$

If $\bar{x}$ is a minimal solution with respect to the intersection of a ball and the feasible set S, one speaks of a *locally minimal solution*.

It is well-known from vector optimization that the set $\min A$ is nonempty in a topological linear space Y with closed ordering cone C, if the set A has a compact section (e.g., see [8, Thm. 6.3,(c)]). A similar result also holds for $\max A$.

The set less order relation $\preccurlyeq_s$ has been introduced to optimization by Kuroiwa (e.g., see [11]; a first publication has been given by Kuroiwa, Tanaka and Ha [12]). Outside the optimization community this notion has been used by Young [17] in algebra, by Nishnianidze [15] in fixed point theory and by Chiriaev and Walster [4] in computer science and interval analysis. The minmax less order relation $\preccurlyeq$ has been defined in [10, Def. 3.5] in order to avoid some drawbacks of the set less order relation $\preccurlyeq_s$. This order relation is more restrictive and from a computational point of view it is even more difficult to treat but it is more realistic in real-world applications.

The set optimization problem (2.1) is formulated without convexity assumptions and, therefore, we cannot expect that one can determine minimal solutions but at best locally minimal solutions.

Based on the order relation $\preccurlyeq$ we now introduce an equivalence relation among sets and a strict relation.

Definition 2.3. Let A, B be subsets of a real linear space Y, which is partially ordered by a convex cone C, with $\min A, \min B, \max A, \max B \neq \emptyset$.

(a) The relation $\sim$ is defined by

$$\begin{aligned} A \sim B \quad :\Longleftrightarrow \quad & (\min A) + C = (\min B) + C,\ (\min A) - C = (\min B) - C, \\ & (\max A) + C = (\max B) + C \text{ and} \\ & (\max A) - C = (\max B) - C. \end{aligned}$$

In this case the sets A and B are called *equivalent*.

(b) Let Y be a topological space and let C be a convex cone in Y. Then the relation $\prec$ is defined by

$$\begin{aligned} A \prec B \quad :\Longleftrightarrow \quad & \exists \text{ an open set } O \subset Y \text{ so that } (\min B) + O \subset (\min A) + C, \\ & (\min A) + O \subset (\min B) - C, \\ & (\max B) + O \subset (\max A) + C, \\ & (\max A) + O \subset (\max B) - C. \end{aligned}$$

Notice that there are equivalent sets which are not equal. It is obvious that the relation $\sim$ is an equivalence relation because it is reflexive, transitive and symmetric. The relation $\prec$ is not an order relation because it is not reflexive although it is transitive. It is some kind of a strict minmax less relation, which is useful for numerical methods.

The following simple lemma is useful for the formulation of various results ([7]); its proof is trivial.

Lemma 2.4. *For nonempty subsets A, B of a real linear space Y and a convex cone $K \subset Y$ it holds*

$$A \subset B + K \iff A + K \subset B + K.$$

Proposition 2.5. *For subsets A, B of a real linear topological space Y, which is partially ordered by a convex cone $C \neq Y$, with $\min A, \min B, \max A, \max B \neq \emptyset$ it holds*

$$A \prec B \implies A \preccurlyeq B \text{ and } A \not\sim B.$$

Proof. Since $A \prec B$, it is obvious that $A \preccurlyeq B$. Moreover, there exists an open set $O \subset Y$ so that

$$(\min B) + O \subset (\min A) + C, \ (\min A) + O \subset (\min B) - C,$$
$$(\max B) + O \subset (\max A) + C, \ (\max A) + O \subset (\max B) - C.$$

With Lemma 2.4 it then follows

$$(\min B) + O + C \subset (\min A) + C, \ (\min A) + O - C \subset (\min B) - C,$$
$$(\max B) + O + C \subset (\max A) + C, \ (\max A) + O - C \subset (\max B) - C.$$

These inclusions imply

$$(\min B) + C \subsetneq (\min A) + C, \ (\min A) - C \subsetneq (\min B) - C,$$
$$(\max B) + C \subsetneq (\max A) + C, \ (\max A) - C \subsetneq (\max B) - C.$$

So, we obtain $A \not\sim B$. □

With the following result the equivalence relation $\sim$ can be characterized in a simpler way.

Proposition 2.6. *Let A, B be subsets of a real linear space Y, which is partially ordered by a convex cone C, with $\min A, \min B, \max A, \max B \neq \emptyset$. Then*

(a) $$(\min A) + C = (\min B) + C \iff \min A = \min B$$

(b) $$(\min A) - C = (\min B) - C \iff \min A = \min B$$

(c) $$(\max A) + C = (\max B) + C \iff \max A = \max B$$

(d) $$(\max A) - C = (\max B) - C \iff \max A = \max B$$

(e) $$A \sim B \iff \min A = \min B \text{ and } \max A = \max B.$$

Proof. (a) The implication "⇐" is trivial. For the proof of the converse implication we obtain with Lemma 2.4

$$\begin{aligned} &(\min A) + C = (\min B) + C \\ &\implies \min A \subset (\min B) + C \\ &\iff \forall\, a \in \min A \ \exists\, b \in \min B, c \in C : a = b + c. \end{aligned} \tag{2.2}$$

Assume that $c \neq 0_Y$. Then we have $a \neq b$ and $b \leq_C a$ where the partial ordering $\leq_C$ is induced by C. Since $a \in \min A$, we conclude $b \notin (\min A) + C = (\min B) + C$. This contradicts the property that $b \in \min B$. Consequently, we have $c = 0_Y$ and $a = b$. The condition (2.2) then implies $\min A \subset \min B$. By renaming we also get $\min B \subset \min A$. So we have $\min A = \min B$.

(b) This proof follows the lines of part (a). For the proof of the implication "⇒" we notice

$$(\min A) - C = (\min B) - C \quad \Longrightarrow \quad \forall\, a \in \min A\ \exists\, b \in \min B, c \in C : a = b - c$$

and the assumption $c \neq 0_Y$ leads to a contradiction. So, we obtain $\min A \subset \min B$ and $\min B \subset \min A$ by renaming. These two inclusions lead to the assertion.
(c),(d) By parts (a) and (b) it holds

$$\begin{aligned}
&(\max A) \pm C = (\max B) \pm C \\
&\iff \quad (-\min(-A)) \pm C = (-\min(-B)) \pm C \\
&\iff \quad (\min(-A)) \mp C = (\min(-B)) \mp C \\
&\iff \quad \min(-A) = \min(-B) \\
&\iff \quad \max A = \max B.
\end{aligned}$$

(e) This assertion immediately follows from the definition of the equivalence relation and the parts (a),...,(d). □

Next, we turn our attention to normed spaces. For arbitrary nonempty subsets A, B of a real normed space $(Y, \|\cdot\|_Y)$ it is well-known that the Hausdorff distance of A and B is given by

$$d_H(A,B) := \max\left\{ \sup_{b\in B} \inf_{a\in A} \|a-b\|_Y,\ \sup_{a\in A} \inf_{b\in B} \|a-b\|_Y \right\}.$$

Based on this distance we now define

$$\psi(A,B) := \max\left\{ d_H(\min A, \min B),\ d_H(\max A, \max B) \right\},$$

if the sets $\min A, \min B, \max A, \max B$ are nonempty.

Now we show that the function ψ is bounded for bounded sets. Recall that the norm of a nonempty subset A of Y is defined by $\|A\| := \sup_{a\in A} \|a\|_Y$.

Lemma 2.7. *Let A, B be subsets of a real normed space $(Y, \|\cdot\|_Y)$, which is partially ordered by a convex cone C, with $\min A, \min B, \max A, \max B \neq \emptyset$. Then it holds*

$$0 \leq \psi(A,B) \leq 2\max\{\|A\|, \|B\|\}.$$

Proof. Let sets $A, B \subset Y$ with $\min A,\ \min B,\ \max A,\ \max B \neq \emptyset$ be arbitrarily chosen. By definition of the function ψ it immediately follows $\psi(A,B) \geq 0$. For an arbitrary $x \in \min A$ we conclude for any $y' \in \min B$

$$\inf_{y\in \min B} \|x-y\|_Y \leq \|x-y'\|_Y \leq \|x\|_Y + \|y'\|_Y \leq 2\max\{\|A\|, \|B\|\}$$

and we obtain

$$\sup_{x\in \min A} \inf_{y\in \min B} \|x-y\| \leq 2\max\{\|A\|, \|B\|\}.$$

Then we also get $d_H(\min A, \min B) \leq 2\max\{\|A\|, \|B\|\}$. Since this inequality also holds for the sets $\max A$ and $\max B$, it follows $\psi(A,B) \leq 2\max\{\|A\|, \|B\|\}$. This completes the proof. □

The essential point is now that the function ψ is closely related to some relations used in this section.

Lemma 2.8. *For subsets A, B of a real normed space $(Y, \|\cdot\|)$, which is partially ordered by a convex cone C, with $\min A, \min B, \max A, \max B \neq \emptyset$ it holds:*

(a) $$A \preccurlyeq B \text{ and } B \preccurlyeq A \implies \psi(A,B) = 0.$$

(b) *If $\min A$, $\max A$, $\min B$ and $\max B$ are compact, then the converse implication in (a) is true.*

(c) $$A \not\sim B \impliedby \psi(A,B) > 0.$$

(d) *If $\min A$, $\max A$, $\min B$ and $\max B$ are compact, then the converse implication in (c) is true.*

(e) *If $\min A$, $\max A$, $\min B$ and $\max B$ are compact, then we have*
$$A \sim B \iff A \preccurlyeq B \text{ and } B \preccurlyeq A.$$

Proof. Let $A, B \subset Y$ with $\min A, \min B, \max A, \max B \neq \emptyset$ be arbitrarily given.
(a) We conclude with Lemma 2.4 and Proposition 2.6

$$\begin{aligned}
&A \preccurlyeq B \text{ and } B \preccurlyeq A\\
&\iff \min A \preccurlyeq_s \min B,\ \max A \preccurlyeq_s \max B,\\
&\qquad \min B \preccurlyeq_s \min A \text{ and } \max B \preccurlyeq_s \max A\\
&\iff \min B \subset (\min A) + C,\ \min A \subset (\min B) - C,\\
&\qquad \max B \subset (\max A) + C,\ \max A \subset (\max B) - C,\\
&\qquad \min A \subset (\min B) + C,\ \min B \subset (\min A) - C,\\
&\qquad \max A \subset (\max B) + C \text{ and } \max B \subset (\max A) - C\\
&\iff (\min B) + C \subset (\min A) + C,\ (\min A) - C \subset (\min B) - C,\\
&\qquad (\max B) + C \subset (\max A) + C,\ (\max A) - C \subset (\max B) - C,\\
&\qquad (\min A) + C \subset (\min B) + C,\ (\min B) - C \subset (\min A) - C,\\
&\qquad (\max A) + C \subset (\max B) + C \text{ and } (\max B) - C \subset (\max A) - C\\
&\iff (\min A) + C = (\min B) + C,\ (\min A) - C = (\min B) - C,\\
&\qquad (\max A) + C = (\max B) + C \text{ and } (\max A) - C = (\max B) - C\\
&\iff \min A = \min B \text{ and } \max A = \max B\\
&\implies d_H(\min A, \min B) = d_H(\max A, \max B) = 0\\
&\iff \psi(A,B) = 0.
\end{aligned}$$

(b) For compact sets $\min A$, $\max A$, $\min B$ and $\max B$ we have

$$d_H(\min A, \min B) = 0 \iff \min A = \min B$$

and

$$d_H(\max A, \max B) = 0 \iff \max A = \max B$$

and, therefore, $d_H(\min A, \min B) = d_H(\max A, \max B) = 0$ implies $\min A = \min B$ and $\max A = \max B$. Then the proof follows the lines of the proof of part (a).

(c) We obtain with Proposition 2.6

$$A \not\sim B \iff \min A \neq \min B \text{ or } \max A \neq \max B$$

$$\begin{aligned} &\Longleftarrow \quad d_H(\min A, \min B) > 0 \text{ or } d_H(\max A, \max B) > 0 \\ &\Longleftrightarrow \quad \psi(A, B) > 0. \end{aligned}$$

(d) This proof follows the lines of the proof of parts (b) and (c).

(e) The assertion follows from (a), (b), (c) and (d). □

The previous lemma already shows that ψ is an important measure for optimality which can be used for numerical methods.

3. A derivative-free descent method

In [9] a descent method has been recently introduced for the determination of minimal solutions of the set optimization problem (2.1) using the set less order relation under certain convexity assumptions. This investigation is now extended to the more complex minmax less order relation and to nonconvex sets $F(x)$ (with $x \in S$). Problems in such a complex setting are difficult to handle so that it makes sense to try to obtain more information about all possible points $x \in S$ with $F(x) \preccurlyeq F(x^0)$ where $x^0 \in S$ is any feasible starting point. Starting from $x^0 \in S$ we do not work with only one descent direction (as in [9]) but with various possible descent directions. The iteration process then produces a tree with the starting point as a root, the iteration points as nodes and the terminal iteration points (nodes without child nodes) as leaves. An illustration of such a rooted tree as result of an iteration process is given in Figure 1. A directed path of this tree, i.e. a sequence of edges (all directed in the same direction) connecting a sequence of nodes, is also illustrated. We speak of a rooted tree descent method, if we determine

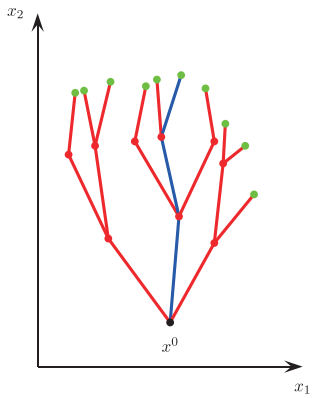

FIGURE 1. Illustration of a rooted tree descent method with starting point $x^0 \in S \subset \mathbb{R}^2$ and one directed path.

all leaves generated by descent directions with $x^0 \in S$ as a starting point. Such a method has a high numerical effort and the advantage that one gets a good overview on locally minimal solutions of the set optimization problem (2.1).

For the investigations in this section we now spezialize Assumption 2.1 to normed spaces and the unconstrained case.

Assumption 3.1. Let $(X, \|\cdot\|_X)$ and $(Y, \|\cdot\|_Y)$ be real normed spaces, let C be a convex cone in Y, and let $F : X \rightrightarrows Y$ be a set-valued map with $\min F(x)$, $\max F(x) \neq \emptyset$ for all $x \in X$.

In the following algorithm we assume that Assumption 3.1 is satisfied.

Algorithm 3.2. (derivative-free rooted tree descent method)

Input: $F : X \rightrightarrows Y$, starting vector $x^0 \in X$, maximal number i_{max} of iterations, number $k \in \mathbb{N}$ of iteration directions, radius $\varepsilon > 0$ of a small ball, smallest positive step length $\delta \ll 1$, step size factor $\mu \in (0, 1)$.

% initialization
choose k directions $d^1, \ldots, d^k$ on the sphere around 0_X with radius ε
nodes := (x^0)
leaves := ()
$i := 0$

% iteration loop
while #(nodes) ≥ 1 and $i \leq i_{max}$ **do**
    $x^i :=$ nodes(1)
    leaf_flag := true
    for $j = 1, \ldots, k$ **do**
        $z^j := x^i + d^j$
        if $z^j \notin$ nodes **then**
            if $F(z^j) \preccurlyeq F(x^i)$ and $F(z^j) \not\sim F(x^i)$ **then** % append this
                % new node
                nodes := (nodes, z^j)
                leaf_flag := false % i.e., x^i cannot be a leaf
            else % refine the step length
                $\lambda := 1$
                while $\lambda \geq \delta$ **do**
                    $\lambda := \mu\lambda$
                    $z^j := x^i + \lambda d^j$
                    if $z^j \notin$ nodes **then**
                        if $F(z^j) \preccurlyeq F(x^i)$ and $F(z^j) \not\sim F(x^i)$ **then**
                            % append this new node
                            nodes := (nodes, z^j)
                            leaf_flag := false % i.e., x^i cannot be a leaf
                            break (and continue after the next
                                end while)
                        end if
                    else
                        leaf_flag := false
                    end if
                end while

```
40              end if
41          else
42              leaf_flag := false
43          end if
44      end for
45      if leaf_flag = true then % append this new leaf
46          leaves := (leaves, x^i)
47      end if
48      nodes := (nodes(2), ...) % i.e., delete the first node
49      i := i + 1
50  end while
51
52  Output: leaves
```

Algorithm 3.2 starts with the selection of possible iteration directions in line 7 and begins at the starting point as root in line 8. The actual iteration process is carried out in the while-loop from line 13 to line 50. x^i (line 14) denotes the i-th iterated node. Then for every direction d^j it is checked whether $x^i + d^j$ (line 17) leads to a decrease von F. If not, with a step size λ, being decreased step by step in line 26, points $x^i + \lambda d^j$ are checked. This is done for $\lambda \geq \delta$ (line 25) in line 27. If a node cannot be improved, it is marked as a leave in line 46. In this way the whole tree with root x^0, including the leaves, is determined.

We begin the investigation of Algorithm 3.2 with a simple result.

Proposition 3.3.

(a) *Algorithm 3.2 is well-defined.*

(b) *Let $x^0 \in X$ be an arbitrary starting point of Algorithm 3.2. For an arbitrary directed path of the rooted tree let the sequence $(x^i)_{i\in\mathbb{N}_0}$ of nodes be generated by Algorithm 3.2 with the property that for every $i \in \mathbb{N}$ the point x^i is a child of the point x^{i-1} in the tree with root x^0. Then we have*

$$F(x^{i+1}) \preccurlyeq F(x^i) \text{ and } F(x^{i+1}) \not\sim F(x^i) \text{ for all } i \in \mathbb{N}_0.$$

Proof. Part (a) is obvious and part (b) follows from the construction of a new node in the "if" part (lines 29–35) in Algorithm 3.2. □

In order to be able to prove convergence of Algorithm 3.2 we follow the theory developed by Torczon et al. [16, 6], but in this special set-valued case we need an appropriate convergence measure. The function ψ introduced in Section 2 can be used as such a convergence measure. The following convergence result concerns an arbitrary directed path of the rooted tree generated by Algorithm 3.2.

Theorem 3.4. *Let $x^0 \in X$ be an arbitrary starting vector of Algorithm 3.2. For an arbitrary directed path of the rooted tree let the sequence $(x^i)_{i\in\mathbb{N}_0}$ of nodes be generated by Algorithm 3.2 with the property that for every $i \in \mathbb{N}$ the node x^i is a child of the node x^{i-1} in the tree with root x^0. Let the sets $F(x)$ be uniformly bounded on this directed path (i.e., there is some $\alpha > 0$ with $\|F(x^i)\| \leq \alpha$ for all $i \in \mathbb{N}_0$). For some $\beta \in (0, 1)$ and a null sequence $(\varepsilon^i)_{i\in\mathbb{N}_0}$ with $\varepsilon^i > 0$ for all $i \in \mathbb{N}_0$*

let the step size rule be used:

While $\psi(F(x^i + \lambda h^i), F(x^i)) \geq \varepsilon^i > 0$ *set* $\lambda := \beta^q \lambda^0$ *for* $q := 0, 1, 2, \ldots$ *(where* λ^0 *and* h^i *denote the initial step length and the descent direction at an iteration point* x^i*, respectively).*

Then it follows

$$\limsup_{i \to \infty} \psi(F(x^{i+1}), F(x^i)) = 0.$$

Proof. Choose an arbitrary directed path of the rooted tree given by the sequence $(x^i)_{i \in \mathbb{N}_0}$ of nodes. Since the sets $F(x)$ are uniformly bounded on the directed path, there is some $\alpha \geq 0$ with $\|F(x^i)\| \leq \alpha$ for all $i \in \mathbb{N}_0$. Then by Lemma 2.7 $0 \leq \psi(F(x^{i+1}), F(x^i)) \leq 2\alpha$ for all $i \in \mathbb{N}_0$. Consequently, the limit superior exists. Assume that $\limsup_{i \to \infty} \psi(F(x^{i+1}), F(x^i)) \neq 0$. Then there exists a subsequence $(x^{i_j})_{j \in \mathbb{N}}$ with $\lim_{j \to \infty} \psi(F(x^{i_j+1}), F(x^{i_j})) =: \beta \neq 0$. Since the function values of ψ are nonnegative by definition, we have $\beta > 0$. Then we obtain

$$\psi(F(x^{i_j+1}), F(x^{i_j})) \geq \frac{\beta}{2} > 0 \text{ for sufficiently large } j \in \mathbb{N}.$$

Since $(\varepsilon^i)_{i \in \mathbb{N}_0}$ is a null sequence, we get

$$\frac{\beta}{2} \geq \varepsilon^{i_j} > 0 \text{ for sufficiently large } j \in \mathbb{N}.$$

Then we obtain

$$\psi(x^{F(i_j+1)}, F(x^{i_j})) \geq \frac{\beta}{2} \geq \varepsilon^{i_j} \text{ for sufficiently large } j \in \mathbb{N}.$$

This is a contradiction to the used step size rule. □

In order to show convergence to some set being minimal in a certain sense we need some special Lipschitz continuity.

Definition 3.5. Let Assumption 3.1 be satisfied. The set-valued map F is called *ψ-Lipschitz continuous* at some $\bar{x} \in X$, if there is a ball $B(\bar{x}, \delta)$ around $\bar{x}$ with radius $\delta > 0$ and some constant $L > 0$ so that

$$\psi(F(x), F(\bar{x})) \leq L\|x - \bar{x}\| \text{ for all } x \in B(\bar{x}, \delta).$$

It is pointed out in [1, Lemma 2.2] that the Hausdorff distance of two compact sets equals the Euclidian norm of the metric difference of the two sets. Since ψ is the maximum of certain Hausdorff distances, there is a close relation of the notion in Definition 3.5 to the standard Lipschitz continuity of set-valued maps [1, Definition 3.1].

Theorem 3.6. *Let the assumptions of Theorem 3.4 be satisfied. For an arbitrary directed path of the rooted tree let the sequence* $(x^i)_{i \in \mathbb{N}_0}$ *of nodes be generated by Algorithm 3.2 with the property that for every* $i \in \mathbb{N}$ *the node* x^i *is a child of the node* x^{i-1} *in the tree with root* x^0*.*

(a) *If the sequence is finite with the final iteration point* x^k *(*$k \in \mathbb{N}$*), then* $F(x^k)$ *is a minimal element of the set system* $\{F(x^0), \ldots, F(x^k)\}$*.*

(b) *For an arbitrary accumulation point $\bar{x}$ of the sequence $(x^i)_{i\in\mathbb{N}_0}$ (i.e. $\bar{x}$ is the limit of a subsequence $(x^{i_j})_{j\in\mathbb{N}}$) let F be ψ-Lipschitz continuous at $\bar{x}$. Then we have*

(i) $\lim\limits_{j\to\infty} \psi(F(x^{i_j}), F(\bar{x})) = 0$.

(ii) *If*

$$F(x^{i_{j+1}}) \prec F(x^{i_j}) \text{ for all } j \in \mathbb{N},$$

then $F(\bar{x})$ is a minimal element of the set system $\big\{F(x^{i_j})_{j\in\mathbb{N}},\, F(\bar{x})\big\}$.

Proof. (a) By Proposition 3.3 we have

$$F(x^k) \preccurlyeq F(x^{k-1}) \preccurlyeq \ldots \preccurlyeq F(x^0)$$

implying that $F(x^k)$ is a minimal element of the set system $\{F(x^0), \ldots, F(x^k)\}$.

(b)(i) Since F is ψ-Lipschitz continuous at $\bar{x}$, there is some $L \geq 0$ so that

$$\psi(F(x^{i_j}), F(\bar{x})) \leq L\|x^{i_j} - \bar{x}\| \text{ for sufficiently large } j \in \mathbb{N}.$$

Because of $\lim\limits_{j\to\infty} x^{i_j} = \bar{x}$ we then conclude

$$\lim_{j\to\infty} \psi(F(x^{i_j}), F(\bar{x})) = 0$$

which has to be shown.

(ii) First we show

$$F(\bar{x}) \prec F(x^{i_j}) \text{ for all } j \in \mathbb{N}. \tag{3.1}$$

Let an index $j \in \mathbb{N}$ be arbitrarily chosen. By the assumption we have $F(x^{i_{j+1}}) \prec F(x^{i_j})$ and, therefore, there exists some $\varepsilon > 0$ with

$$\begin{aligned}
\big(\min F(x^{i_j})\big) + B(0_Y, \varepsilon) &\subset \big(\min F(x^{i_{j+1}})\big) + C,\\
\big(\min F(x^{i_{j+1}})\big) + B(0_Y, \varepsilon) &\subset \big(\min F(x^{i_j})\big) - C,\\
\big(\max F(x^{i_j})\big) + B(0_Y, \varepsilon) &\subset \big(\max F(x^{i_{j+1}})\big) + C,\\
\big(\max F(x^{i_{j+1}})\big) + B(0_Y, \varepsilon) &\subset \big(\max F(x^{i_j})\big) - C.
\end{aligned}$$

Since $\lim\limits_{j\to\infty} \psi(F(x^{i_j}), F(\bar{x})) = 0$ by part (i), for an arbitrary $\delta > 0$ there exists some $j' \in \mathbb{N}$ with $j' \geq j+1$ and $\psi(F(x^{i_{j'}}), F(\bar{x})) \leq \delta$. This implies

$$\begin{aligned}
d_H(\min F(x^{i_{j'}}), \min F(\bar{x})) &\leq \delta,\\
d_H(\max F(x^{i_{j'}}), \max F(\bar{x})) &\leq \delta.
\end{aligned}$$

Then we obtain

$$\begin{aligned}
\big(\min F(x^{i_j})\big) + B(0_Y, \varepsilon) &\subset \big(\min F(x^{i_{j+1}})\big) + C\\
&\subset \big(\min F(x^{i_{j'}})\big) + C\\
&\subset (\min F(\bar{x})) + B(0_Y, \delta) + C
\end{aligned} \tag{3.2}$$

and

$$\big(\max F(x^{i_j})\big) + B(0_Y, \varepsilon) \subset (\max F(\bar{x})) + B(0_Y, \delta) + C. \tag{3.3}$$

The inclusions

$$\begin{aligned}\min F(\bar{x}) &\subset (\min F(\bar{x})) - C \\ &\subset \big(\min F(x^{i_{j'}})\big) + B(0_Y, \delta) - C \\ &\subset \big(\min F(x^{i_{j+1}})\big) + B(0_Y, \delta) - C\end{aligned}$$

imply

$$\begin{aligned}(\min F(\bar{x})) + B(0_Y, \varepsilon) &\subset \underbrace{\big(\min F(x^{i_{j+1}})\big) + B(0_Y, \varepsilon)}_{\subset \big(\min F(x^{i_j})\big) - C} + B(0_Y, \delta) - C \\ &\subset \big(\min F(x^{i_j})\big) + B(0_Y, \delta) - C. \qquad (3.4)\end{aligned}$$

With the same arguments we get

$$(\max F(\bar{x})) + B(0_Y, \varepsilon) \subset \big(\max F(x^{i_j})\big) + B(0_Y, \delta) - C. \qquad (3.5)$$

The inclusions (3.2), (3.4), (3.3) and (3.5) hold for all $\delta > 0$ and consequently we obtain

$$\begin{aligned}\big(\min F(x^{i_j})\big) + B\Big(0_Y, \frac{\varepsilon}{2}\Big) &\subset (\min F(\bar{x})) + C, \\ (\min F(\bar{x})) + B\Big(0_Y, \frac{\varepsilon}{2}\Big) &\subset \big(\min F(x^{i_j})\big) - C, \\ \big(\max F(x^{i_j})\big) + B\Big(0_Y, \frac{\varepsilon}{2}\Big) &\subset (\max F(\bar{x})) + C, \\ (\max F(\bar{x})) + B\Big(0_Y, \frac{\varepsilon}{2}\Big) &\subset \big(\max F(x^{i_j})\big) - C.\end{aligned}$$

This means that $F(\bar{x}) \prec F(x^{i_j})$, and the inequality (3.1) is shown.
The inequality (3.1) immediately implies that $F(\bar{x})$ is a minimal element of the set system $\big\{F(x^{i_j})_{j\in\mathbb{N}}, F(\bar{x})\big\}$. □

Notice that it is required in part (b),(ii) of the preceding theorem that Algorithm 3.2 generates sets being strictly ordered. Proposition 2.5 clarifies this point.

4. Numerical results

Although we investigate nonconvex sets in this paper, we still need a certain structure of these sets for the implementation of Algorithm 3.2. As a possible structure we can assume that these sets are piecewise starshaped with respect to certain elements.

Definition 4.1. A nonempty subset A of a real linear space is called *piecewise starshaped* iff there are finitely many nonempty subsets $A_1, \ldots, A_r$ of A (with $r \in \mathbb{N}$) so that $A = \bigcup_{i=1}^{r} A_i$ and every subset A_i (with $i \in \{1, \ldots, r\}$) is starshaped with respect to some $\hat{y}^i \in A_i$, i.e.

$$\lambda y + (1-\lambda)\hat{y}^i \in A_i \text{ for all } \lambda \in [0,1] \text{ and all } y \in A_i.$$

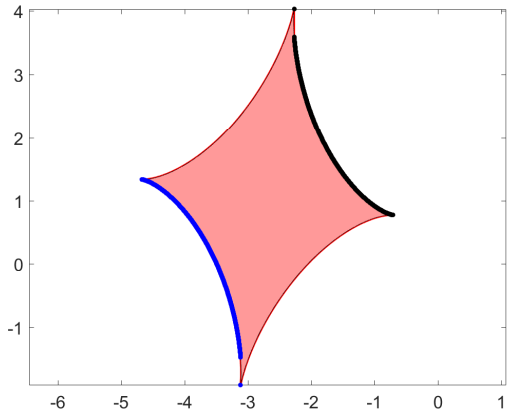

FIGURE 2. Set $F(2,3)$ with minimal and maximal elements in Example 4.2.

Even though every nonempty convex set is also starshaped with respect to each of its elements, starshaped sets are nonconvex, in general. Since a piecewise starshaped set is the union of finitely many starshaped subsets, even highly nonconvex sets can be considered. Such a set may also be non-connected so that we work with a very general class of sets.

As another possible structure we can assume that the boundary of nonconvex sets is parameterized by some parameters. The following examples work with starshaped sets with a parameterization of the boundary. The computations are carried out with Matlab R2014a on a Fujitsu Celsius R570-2 with Nvidia's GPU Tesla C2075. Simple tests with this GPU have shown for the computation of all minimal and maximal elements of a discrete set that speed-ups of 32 are possible.

Example 4.2. Let $F : \mathbb{R}^2 \rightrightarrows \mathbb{R}^2$ be a set-valued map where for every $x \in \mathbb{R}^2$ the boundary of the set $F(x)$ is given by

$$\partial F(x) = \left\{ \left(\begin{array}{l} e^{0.5x_1} \cos x_2 + x_1 \cos x_2 \cos^3 \varphi - x_2 \sin x_2 \sin^3 \varphi \\ e^{0.05x_2} \sin x_1 + x_1 \sin x_2 \cos^3 \varphi + x_2 \cos x_2 \sin^3 \varphi \end{array} \right) \in \mathbb{R}^2 \;\middle|\; \varphi \in [0, 2\pi) \right\}.$$

The set $F(x)$ (with $x \in \mathbb{R}^2$) is a shifted, warped and rotated asteroid; an example of such an asteroid is illustrated in Figure 2. Obviously, asteroids are nonconvex sets. In $\mathbb{R}^2$ let the partial ordering be given in a componentwise sense, i.e. $C := \mathbb{R}^2_+$. Starting at the point $x^0 = (3,3)$ Figure 3 illustrates a rooted tree generated by Algorithm 3.2. Here $k := 40$ iteration directions are used, and the Euclidean norm of these vectors is chosen as $\varepsilon := 0.4$. The boundary of the considered sets is discretized by 1,000 points. The step size factor, which decreases the step length, is set to $\mu := 0.5$.

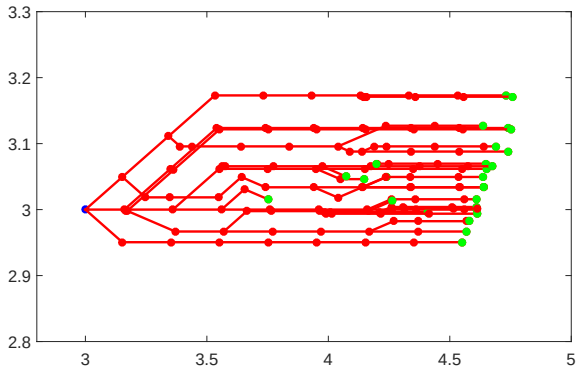

FIGURE 3. Illustration of a rooted tree generated by Algorithm 3.2 for Example 4.2 with starting point $(3, 3)$.

Now we turn our attention to sets in 3 dimensions. This is possible, if one makes computations in parallel.

Example 4.3. Let $F : \mathbb{R}^2 \rightrightarrows \mathbb{R}^3$ be a set-valued map where for every $x \in \mathbb{R}^2$ the boundary of the set $F(x)$ is given by

$$\begin{aligned}
&\partial F(x) \\
&= \left\{ g(x) + (x_1^2 + x_2^4) \begin{pmatrix} (2.1 + x_1^2 + \cos\frac{\varphi_1}{2} \sin\varphi_2 - \sin\frac{\varphi_1}{2} \sin 2\varphi_2) \cos\varphi_1 \\ (2.1 + x_1^2 + \cos\frac{\varphi_1}{2} \sin\varphi_2 - \sin\frac{\varphi_1}{2} \sin 2\varphi_2) \sin\varphi_1 \\ \sin\frac{\varphi_1}{2} \sin\varphi_2 + \cos\frac{\varphi_1}{2} \sin 2\varphi_2 \end{pmatrix} \right. \\
&\qquad \left. \in \mathbb{R}^3 \;\middle|\; \varphi_1, \varphi_2 \in [0, 2\pi) \right\}
\end{aligned}$$

with

$$\begin{aligned}
g(x) \;:=\; 100 \Bigg[&\begin{pmatrix} -x_1 \\ x_1 + x_2^2 \\ -x_1 \end{pmatrix} + 100 \max\{0, x_1^2 - x_2\} \\
&+ 100 \max\{0, x_1 + 2x_2 - 3\} \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \Bigg].
\end{aligned}$$

The set $F(x)$ (with $x \in \mathbb{R}^2$) is a shifted and scaled set being called the "figure 8" immersion of the Klein bottle, and it is illustrated in Figure 4 for the starting point $x^0 := (-2, 2)$. Starting at x^0 Figure 5 illustrates a rooted tree generated by Algorithm 3.2 with $k := 10$ iteration directions. The boundary of the considered sets is discretized by $10,000$ points. Other parameters are chosen as in Example 4.2. The point $x^* := (-0.0044902, -0.070108)$ is one of the leaves of the rooted tree and the set $F(x^*)$ is illustrated in Figure 6.

The following example shows that parallel computation is the base even for problems with four dimensional sets.

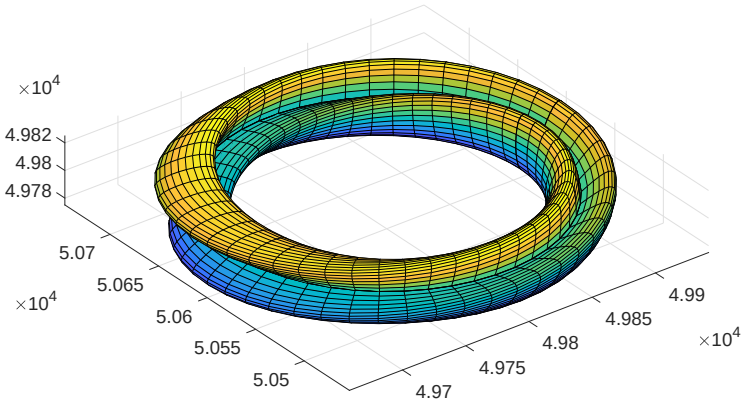

FIGURE 4. Illustration of $F(-2, 2)$ in Example 4.3.

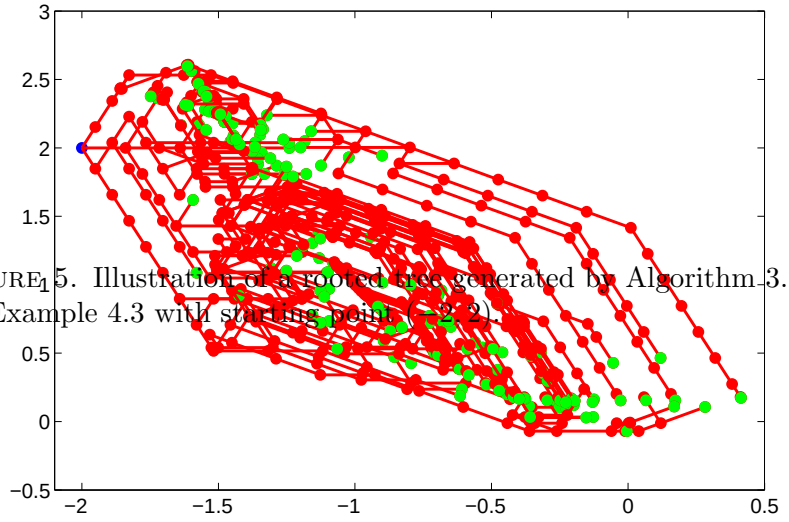

FIGURE 5. Illustration of a rooted tree generated by Algorithm 3.2 for Example 4.3 with starting point $(-2, 2)$.

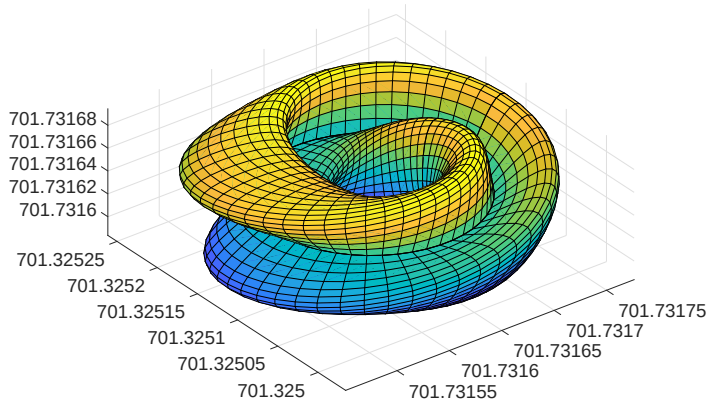

FIGURE 6. Illustration of $F(x^*)$ in Example 4.3.

Example 4.4. Let $F : \mathbb{R}^2 \rightrightarrows \mathbb{R}^4$ be a set-valued map where for every $x \in \mathbb{R}^2$ the boundary of the set $F(x)$ is given by

$$\begin{aligned} \partial F(x) \quad = \quad & \left\{ g(x) + \left(x_1 + \frac{1}{10} \sin 20\varphi_1 \sin 20\varphi_2 \right) \begin{pmatrix} \cos\varphi_1 \\ \sin\varphi_1 \cos\varphi_2 \\ \sin\varphi_1 \sin\varphi_2 \cos\varphi_3 \\ \sin\varphi_1 \sin\varphi_2 \sin\varphi_3 \end{pmatrix} \right. \\ & \left. \in \mathbb{R}^4 \;\middle|\; \varphi_1, \varphi_2 \in [0, 2\pi), \varphi_3 \in [0, \pi) \right\} \end{aligned}$$

with

$$g(x) := 100(x_1^2 + x_2^2 - 9)^2 \begin{pmatrix} |x_1| \\ |x_2| \\ |x_1 + x_2| \\ |x_2| \end{pmatrix}.$$

The projection of the set $F(x)$ (with $x \in \mathbb{R}^2$) to the y_1-y_2-y_3 coordinate system is for $\varphi_3 = 0$ some kind of a hyperball with "oscillating radius" in $\mathbb{R}^3$ (compare Figure 7). Figure 8 illustrates a rooted tree generated by Algorithm 3.2 with the starting vector $x^0 = (1, 1)$ and $k := 11$ iteration directions. The boundary of the considered sets is discretized by 103,823 points. Such a high discretization leads to a high elapsed time of about 7 days and 23 hours and 12 minutes. The other parameters are chosen as in Example 4.2.

In Figure 8 a part of the circle with center $(0, 0)$ and radius 3 is also drawn. It turns out that most of the leaves are located very close to this circle. Only 6 leaves have a certain distance to this circle. At these points it was not possible to get an improvement with only 11 iteration directions. Since this circle plays an important role in the translation $g(x)$ in the definition of the boundary of $F(x)$ (with $x \in \mathbb{R}^2$),

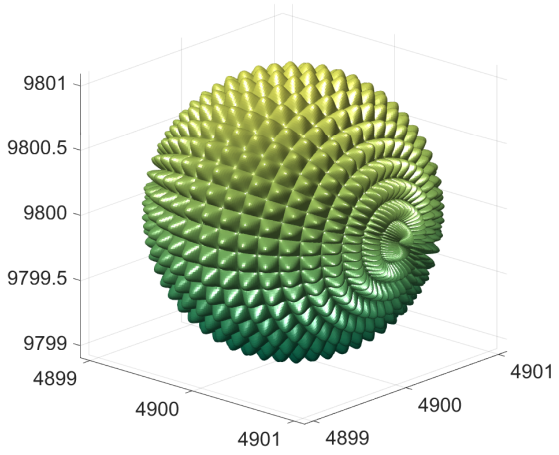

FIGURE 7. Illustration of the projection of $F(1,1)$ for $\varphi_3 = 0$ in $\mathbb{R}^3$ in Example 4.4.

it seems that this circular arc describes solutions of this special set optimization problem.

Algorithm 3.2 computes 436 nodes; most of them are needed for a refinement of the step length. This leads to the good approximation of the circular arc by

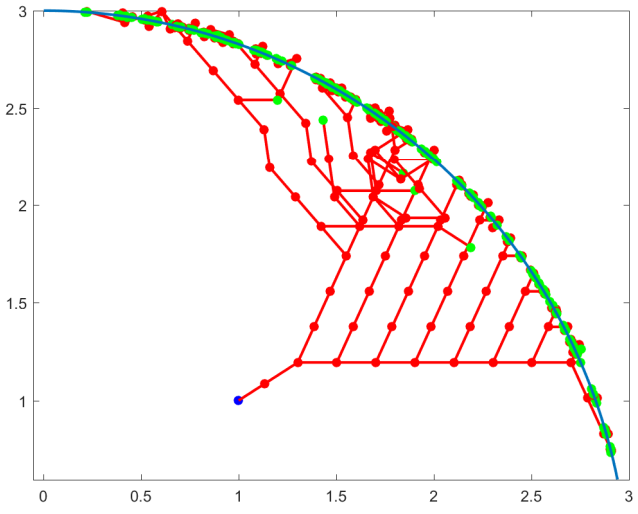

FIGURE 8. Illustration of a rooted tree generated by Algorithm 3.2 for Example 4.4 with the starting point $(1,1)$ together with the circle with center $(0,0)$ and radius 3.

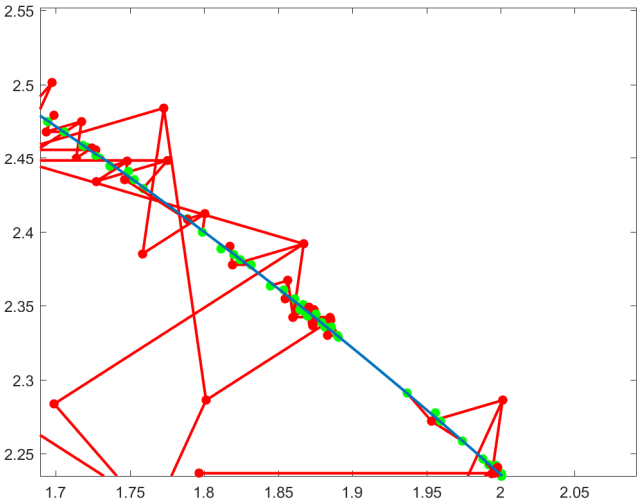

FIGURE 9. Enlarged part of Figure 8.

the computed leaves. If one enlarges a part of this circuler arc, one can see the approximation behaviour of this algorithm in Figure 9.

If one works only with 1,000 discretization points of the considered sets, 403 nodes are computed with an elapsed time of about 1 hour and 6 minutes. The generated rooted tree is very similar to the illustration in Figure 8. It is interesting to observe that although the finer discretization is 103 times as high as the coarser discretization, the elapsed time is about 174 times longer than in the simple case. Although the complexity increases strongly, the increase of the elapsed time is moderate because parts of Algorithm 3.2 work in parallel.

5. Conclusion

The method of this paper can also be adapted to problems using simpler order relations, like the set less relation or its lower and upper versions. It seems that parallel computing is the key for the numerical solution of set optimization problems. Although GPUs are very helpful in set optimization, the numerical investigations could also be extended to multicore machines.

Acknowledgement
The author thanks E. Köbis for discussions on numerical aspects in a very early stage of this paper and T.X.D. Ha for discussions on parts of this paper. He is also indebted to an anonymous referee for constructive comments on a first version of this paper.

Appendix

Although computation has been done with Matlab R2014a, computational expensive parts can be carried out in parallel using a graphics processing unit (GPU). Listing 1 gives a simple example of a CUDA C program code for the determination of minimal and maximal elements of sets in $\mathbb{R}^2$. The points of the discretized boundary of a set are contained in the input array $(y\_1[\,], y\_2[\,])$ and the input point $(y1, y2)$ is the reference point which is checked for minimality and maximality. Output variables are *sum_min* and *sum_max* with the following meaning: If

$$sum\_min = \text{ number of discretized boundary points,}$$

then $(y1, y2)$ is a minimal point of the set of these points. With *sum_max* one gets a similar result with respect to maximality.

Listing 1. Computation of minimal and maximal points on Nvidia's GPUs

```
//
//  CUDA-Kernel: Determination of minimal and maximal points
//
__global__ void Min_Max(const float *y_1, const float *y_2,
                        const float y1, const float y2,
                        int *sum_min, int *sum_max,
                        const int numElements,
                        const int ThreadsPerBlock,
                        const int LastBlockID,
                        const int ThreadsPerLastBlock )
{   __shared__ int temp_min[448];
    __shared__ int temp_max[448];
    int i = blockDim.x * blockIdx.x + threadIdx.x;
    if (i < numElements)
    {   if ((y_1[i] <= y1 && y_2[i] < y2) ||
            (y_1[i] < y1 && y_2[i] <= y2))
           { temp_min[threadIdx.x] = 0; }
        else
           { temp_min[threadIdx.x] = 1; }
        if ((y_1[i] >= y1 && y_2[i] > y2) ||
            (y_1[i] > y1 && y_2[i] >= y2))
           { temp_max[threadIdx.x] = 0; }
        else
           { temp_max[threadIdx.x] = 1; }
    }
    __syncthreads();
    if (0 == threadIdx.x && blockIdx.x < LastBlockID)
    {   int sum1 = 0;
        int sum2 = 0;
        for( int j = 0; j < ThreadsPerBlock; j++ )
           { sum1 += temp_min[j];
             sum2 += temp_max[j]; }
        atomicAdd( sum_min , sum1 );
        atomicAdd( sum_max , sum2 );
```

```
    }
    if (0 == threadIdx.x && blockIdx.x == LastBlockID)
    {
        int sum1 = 0;
        int sum2 = 0;
        for( int j = 0; j < ThreadsPerLastBlock; j++ )
           { sum1 += temp_min[j];
             sum2 += temp_max[j]; }
        atomicAdd( sum_min , sum1 );
        atomicAdd( sum_max , sum2 );
    }
}
```

If sets in $\mathbb{R}^2$ are starshaped with respect to some reference point and if boundary points are given by radial functions, the Matlab *arrayfun* function transforms the radial function written in Matlab to a CUDA C version being carried out on the GPU. Listing 2 illustrates this point where t is a discretization of the interval $[0, 2\pi]$ and x is the actual parameter vector.

Listing 2. Computation of boundary points on Nvidia's GPUs

```
y_hat = ReferencePoint(x);
t_gpu = gpuArray(t);
[y_1_gpu,y_2_gpu] ...
  = arrayfun(@RadialFunction,t_gpu,x(1),x(2),y_hat(1), ...
             y_hat(2));
y_1 = gather(y_1_gpu);
y_2 = gather(y_2_gpu);
```

A concrete formulation of the radial function of Example 4.2 is given in Listing 3. The radial function values are then determined on the GPU whereas the reference point is computed on the CPU.

Listing 3. Evaluation of radial functions on Nvidia's GPUs

```
% radial function on GPUs
function [y_1,y_2] = RadialFunction(t,x1,x2,y1_hat,y2_hat)
    y_1=y1_hat+x1.*cos(x2).*(cos(t).^3)-x2.*sin(x2) ...
        .*(sin(t).^3);
    y_2=y2_hat+x1.*sin(x2).*(cos(t).^3)+x2.*cos(x2) ...
        .*(sin(t).^3);
end
% reference point on CPU
function [y_hat] = ReferencePoint(x)
    y_hat=[exp(0.5*x(1))*cos(x(2)) ...
           exp(0.05*x(2))*sin(x(1))];
end
```

References

[1] R. Baier and E. Farkhi, *Regularity of set-valued maps and their selections through set differences: Part 1: Lipschitz continuity*, Serdica Math. J. **39** (2013), 365–390.

[2] M. Bischoff and J. Jahn, *Economic objectives, uncertainties and decision making in the energy sector*, J. Bus. Econ. **86** (2016), 85–102.

[3] M. Bischoff, J. Jahn and E. Köbis, *Hard uncertainties in multiobjective layout optimization of photovoltaic power plants*, Optimization **66** (2017), 361–380.

[4] A. Chiriaev and G. W. Walster, *Interval Arithmetic Specification*, Technical Report, 1998.

[5] A. R. Conn, K. Scheinberg and L. N. Vicente, *Introduction to Derivative-Free Optimization*, MPS-SIAM Series on Optimization, SIAM, Philadelphia, 2009.

[6] E. D. Dolan and R. M. Lewis and V. Torczon, *On the local convergence of pattern search*, SIAM J. Optim. **14** (2003), 567–583.

[7] T. X. D. Ha, *private communication*, 2015.

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

[9] J. Jahn, *A derivative-free descent method in set optimization*, Comput. Optim. Appl. **60** (2015), 393–411.

[10] J. Jahn and T. X. D. Ha, *New Order Relations in Set Optimization*, J. Optim. Theory Appl. **148** (2011), 209–236.

[11] D. Kuroiwa, *Natural criteria of set-valued optimization*, manuscript, Shimane University, Japan, 1998.

[12] D. Kuroiwa, T. Tanaka and X. T. D. Ha, *On cone convexity of set-valued maps*, Nonlinear Anal. **30** (1997), 1487–1496.

[13] S. Limmer, D. Fey and J. Jahn, *GPU implementation of a multiobjective search algorithm*, Positivity **16** (2012), 397–404.

[14] N. Neukel, *Order Relations of Sets and its Application in Socio-Economics*, Appl. Math. Sci. (Ruse) **7** (2013), 5711–5739.

[15] Z. G. Nishnianidze, *Fixed points of monotonic multiple-valued operators (in Russian)*, Bull. Georgian Acad. Sci. **114** (1984), 489–491.

[16] V. Torczon, *On the convergence of pattern search algorithms*, SIAM J. Optim. **7** (1997), 1–25.

[17] R. C. Young, *The algebra of many-valued quantities*, Math. Ann. **104** (1931), 260–290.

Manuscript received September 6 2017
revised December 13 2017

J. Jahn
Department Mathematik, Universität Erlangen-Nürnberg, Cauerstr. 11, 91058 Erlangen, Germany
E-mail address: johannes.jahn@fau.de