Karlsruhe Institute of Technology

Department of Economics and Management
Institute of Applied Informatics and Formal Description Methods (AIFB)

Diploma thesis

Robustness in Multiobjective Optimization

of

Joscha Kaiser
Matr. Nr.: 1480371
Course of studies: Business Mathematics

Handover date:
29.05.2015

Supervision: Dr. rer. nat. Pradyumn Kumar Shukla

Erklärung

Ich versichere wahrheitsgemäß, die Arbeit selbstständig angefertigt und alle benutzten Hilfsmittel und Quellen vollständig angegeben zu haben, die wörtlich oder inhaltlich übernommenen Stellen als solche kenntlich gemacht zu haben und die Satzung des KIT zur Sicherung guter wissenschaftlicher Praxis beachtet zu haben.

Datum *Name*

Contents

1 Preface

1.1 Introduction

Most practical optimization problems are of multiobjective nature. They usually represent a decision problem which is characterized by multiple criteria such as time, reliability, efficiency, safety, cost, and so on. Since these objectives typically are conflicting under the given problem, the existence of a unique optimal solution is very unlikely but rather many or even infinitely many decisions are suitable.
Another common characteristic of real-world optimization problems is, that they have to deal with some kind of perturbations or uncertainties, which may occur due to unforeseeable future events, estimation or computational errors. Those perturbations can effect an optimization problem in all of its components, such as input and output data as well as the constraints and also the decision maker's preferences may vary under uncertainties. For an optimization problem contaminated with uncertain data it is very important to estimate the effects of this uncertainty and so it is necessary to evaluate how sensitive an optimal solution is to perturbations of the infected data.
One way to deal with such kind of problems is the field of robust multiobjective optimization.

In this thesis we regard multiobjective optimization problems which are searching for efficient solutions based on an ordering relation induced by an arbitrary convex cone in the objective space. We develop a new definition of robustness for multiobjective optimization problems suffering under perturbations within the ordering inducing cone and present an evolutionary algorithm able to find such cone robust efficient solutions. This algorithm also assigns a degree to each found efficient solution, which is measuring the solutions maximal resistance capability against increasing perturbations within the ordering inducing cone. In the second part we regard optimization problems suffering under uncertain input data concerning the objective function. We present the connection between such so called uncertain optimization problems and set-valued optimization problems and introduce seven ordering relations on the power set representing the objective space. Also we present an evolutionary algorithm able to solve a special class of set-valued optimization problems equivalent to uncertain optimization problems. This opens a new field of application for evolutionary algorithms on the growing research unit of set-valued optimization problems. In general an evolutionary algorithm creates a population, i.e. a set of random solutions for a given multiobjective optimization problem and efficiently evolves this population

over a predefined amount of iterations or until some other kind of stopping criteria is reached. Recently, evolutionary algorithms become increasingly popular. Next to a lot of theoretical studies, they are as well applicable to many real world problems on topics as chemistry, mechanical engineering, civil engineering etc. [6], [41]. Reasons for this increasing popularity are given by the many advantages evolutionary algorithms bring along in comparison to conventional optimization techniques. One of the most important might be, that evolutionary algorithms in general do not need any derivatives of the function to be optimized because of their stochastic nature and their consideration of an optimization problem as a black box. For real world problems an extraction of gradient information often is a very tough task or even impossible [29], thus derivative-free optimization techniques are often more applicable. Other important advantages to ordinary gradient-based optimization methods are their random operators and their population-basis. These prevent evolutionary algorithms from getting stuck in local optima and let them evolve a whole population of solutions representing the complete efficient front of a multiobjective optimization problem, in one single run. In this way, the decision maker is provided with the opportunity to get to know the shape of the efficient front and the efficient set and is provided with the opportunity to choose of a broad variety of optimal solutions.

This diploma thesis is structured as follows. In section 1.2 we will give a short literature review of different robustness approaches existent for multiobjective optimization problems. Section 1.3 covers the basic notation on binary relations, cones and deterministic multiobjective optimization and presents a special class of cones, which is of high importance to the computational components of this thesis. In section 2 we will introduce the new concept of cone robustness, which covers the topic of uncertainties within the ordering inducing cone. In section 2.1 we will present and evaluate an evolutionary algorithm able to search for such robust solutions and we will end the chapter with a conclusion in section 2.2, pointing out open questions and aspects of future work. In section 3 we will discuss uncertain multiobjective optimization problems and we will present an introduction into the field of set-valued optimization in section 3.1. Furthermore, section 3.2 covers the definition of robustness for uncertain optimization problems and illustrates the connection between set-valued optimization and uncertain optimization. In section 3.3 we will present an evolutionary algorithm able to run on set-valued or uncertain mutliobective optimization problems respectively and in section 3.4 we will end this chapter with a conclusion as well. Finally we will finish the thesis with some closing remarks in section 4.

1.2 Literature Review

There are a lot of different ways of how to model uncertainties in multiobjective optimization problems and how to test solutions on robustness, presented in the literature existent on the topic of robustness in multiobjective optimization. In the following we are going to introduce some of the most famous works on this subjects as well as the work which was most influential for this thesis.

As pioneers on the topic of robustness in multiobjective optimization often K. Dep and H. Gupta's work [8] from 2006 is stated. They primarily focus on perturbations in the decision space and define two types of robust solutions. The first one is called robustness of type I. This definition is an extension of an averaging approach for singleobjective optimization problems[43, Tsusui and Gosh, 1977], [4, Branke, 1998] and defines a solution as robust iff it is a solution to minimizing the so called mean effective objective function which is the vector-valued function, consisting of the mean values of each original component. The second robustness approach is called robustness of type II, here Dep and Gupta minimize over the original objectives but add a constraint, which limits the extent of functional change under perturbations in the decision space to a user-defined threshold. For both methods the authors always consider a solution as minimal if it is efficient w.r.t. to the definition of efficiency initial introduced to applied sciences by Edgeworth (1881) [11] and Pareto (1906)[39], respectively. Note that in many papers on the topic of multiobjective optimization, efficient solutions are denoted as Pareto optimal solutions. Since Edgeworth's notions are very close to the ones given by Pareto it is historically appropriate to speak of Edgeworth-Pareto optimal solutions, as for example suggested by Stadler [38] and we refer to [38] as well for a more detailed biographical survey of history and developments in multiobjective optimization. Furthermore we have to remark, that there does not exist a unique definition for efficiency in literature as for instance pointed out by M. Ehrgott in [13], some authors use optimal, non-dominated or minimal for what we will call efficient.

In the work of L. T. Bui, H. A. Abbass, M. Barlow and A. Bender [1] from 2010 the authors measure robustness in relation to perturbations of the decision makers attitude to risk in two different ways. First they introduce the so called dominance robustness, which is the ability of an Edgeworth-Pareto optimal solution to stay efficient, when it is perturbed in the decision space. Secondly they present the preference robustness as the minimum transition cost of an Edgeworth-Pareto optimal solution in its decision space, when it

is perturbed in the objective space. Both definitions of robustness are designed in order to hedge against uncertainties in the decision maker's attitude to risk, which are significant due to the fact that the decision maker often changes over the period of decision making. Our notion of robustness introduced in section 2 as well is based on the concern of uncertainties within the decision makers attitude to risk or his preferences in general. But instead of regarding perturbations within the objective or decision space we directly approach the binary relation on the objective space which induces the definition of domination between two solutions and hence represents the decision makers preferences.

At last we want to introduce the work of J. Ide [17] from 2014, which gave the thought provoking impulse for section 3. Here the author presents five papers which are joint work with E. Köbis, A. Schöbel, et al. on the topic of robustness in uncertain multiobjective optimization [12], [20], [18], [19], [21]. Uncertain optimization problems regard possible scenarios or realizations concerning the objective function, which leads towards a family of different deterministic multiobjective optimization problems. Many different concepts of what is considered to be a robust solution have been presented within the work of Ide et al. In the first paper the authors extend the concept of minmax robustness from single objective optimization problems to multiobjective optimization problems. Minmax robustness was initially introduced by Soyster [37] in 1973 and addresses the idea to find solutions which are feasible in every scenario and thus hedges against the worst case of all scenarios. In the other papers presented, the authors introduce various additional concepts of what is considered to be a robust solution to an uncertain multiobjective optimization problem and present a connection between multiobjective set-valued optimization and uncertain multiobjective optimization in [18]. This connection is studied in [19] furthermore and some of the robustness concepts are extended to general spaces and cones which was crucial for our work on this subject in section 3, as we are working with general cones as well. Finally the authors present an application of the concept of minmax robust optimization onto a real world problem out of the wood cutting industry in the last paper ([21]).

1.3 Notation

First we will have to give the most basic notations necessary in this work, in order to guarantee the integrity of all results and proofs to follow.

Throughout this thesis we are working with *real multidimensional linear*

spaces $\mathbb{R}^k$ with $k \in \mathbb{N} \setminus \{1\}$ being a natural number bigger than one. If not denoted differently we will work with the *Euclidean norm* $\|\cdot\|_2$ and the *canonical scalar product* $\langle \cdot, \cdot \rangle$ in $\mathbb{R}^k$. For $\epsilon \in \mathbb{R}, \epsilon > 0$ and $\bar{y} \in \mathbb{R}^k$, a *ϵ-neighborhood* of $\bar{y}$ is labeled as $B_\epsilon(\bar{y}) := \{y \in \mathbb{R}^k \,|\, \|y - \bar{y}\|_2 < \delta\}$. Given an arbitrary matrix $A \in \mathbb{R}^{l \times m}$, with $l, m \in \mathbb{N}$ we will denote its *rank* as $rg(A)$ and its *transpose* as $A^\mathsf{T} \in \mathbb{R}^{m \times l}$. For an arbitrary countable set S the *cardinality* is given by $|S|$ and for a subset Y of $\mathbb{R}^k$ the *linear span* is denoted as $lin(Y)$. The *dimension* $\dim(Y)$ of Y is defined to be $\dim(Y) := \dim(lin(Y))$ the dimension of the linear space spanned by Y. The set $Y \subseteq \mathbb{R}^k$ is called *closed*, iff the limit of each convergent sequence in Y lies in Y as well and an element $y \in Y$ is said to be an *interior point* if there exists an $\epsilon > 0$, s.t. $B_\epsilon(y) \subseteq Y$. The *interior* of Y contains all interior points of Y and is denoted as $int(Y) := \{y \in Y \,|\, y \text{ is an interior point of } Y\}$. Furthermore we will denote $id^k : \mathbb{R}^k \to \mathbb{R}^k,\ x \mapsto x$, as the *identity function* in $\mathbb{R}^k$ and $\mathscr{P}(\mathbb{R}^k) := \{A \subseteq \mathbb{R}^k \,|\, A \neq \emptyset\}$ as the *power set* of $\mathbb{R}^k$ without the empty set. At last given two subsets A, B of $\mathbb{R}^k$ we will denote the *Minkowski sum* as $A + B := \{a + b \,|\, a \in A, b \in B\}$ as well as $A - B := \{a - b \,|\, a \in A, b \in B\}$.

Now we can present the basic notations on binary relations and cones which are of importance for this thesis.

1.3.1 Binary Relations and Cones

First let us recall a few important properties of binary relations.

Definition 1.1
A binary relation ρ on an arbitrary set Y is said to be:

1. reflexive *iff for all $y \in Y$ $y \,\rho\, y$ holds true,*
2. irreflexive *iff for all $y \in Y$ $y \,\rho\, y$ does not hold true,*
3. antisymmetric *iff for all $y, z \in Y$ with $y \,\rho\, z$ and $z \,\rho\, y$ follows that $y = z$,*
4. transitive *iff for all $y, z, w \in Y$ with $y \,\rho\, z$ and $z \,\rho\, w$ follows that $y \,\rho\, w$.*
5. total *iff for all $y, z \in Y$ either $y \,\rho\, z$ or $z \,\rho\, y$ holds true.*

Definition 1.2
A binary relation ρ on an arbitrary set Y is called a:

1. strict partial order *iff it is irreflexive and transitive,*
2. pre-order *iff it is reflexive and transitive,*

3. partial order *iff it is an antisymmetric pre-order,*

4. total order *iff it is total and a partial order.*

We call a binary relation ρ on Y an ordering relation *on Y iff it is a strict partial order, pre-order, partial order or a total order.*

The following lemma will be of importance to the various kinds of notations for domination we are working with in this thesis.

Lemma 1.3
Let Y be an arbitrary set, $\preccurlyeq$ a pre-order on Y and let $\prec$ be defined as follows. For all $y, z \in Y$

$$x \prec y \ :\Longleftrightarrow \ x \preccurlyeq y \ \text{ and } \ \neg(y \preccurlyeq x).$$

Then $\prec$ is a strict partial order on Y.

Proof:

Irreflexivity: Assume $\prec$ to be reflexive, then there exists a $y \in Y$ with $y \prec y$ which is equivalent to

$$y \preccurlyeq y \ \text{ and } \ \neg(y \preccurlyeq y).$$

Hence $\prec$ has to be irreflexive.

Transitivity: Let $y, z, w \in Y$ with $y \prec z \prec w$, which means

$$y \preccurlyeq z \ \text{ and } \ z \preccurlyeq w, \tag{1}$$
$$\neg(z \preccurlyeq y) \ \text{ and } \ \neg(w \preccurlyeq z). \tag{2}$$

Since $\preccurlyeq$ is a pre-order and hence transitive it follows with (1) that $y \preccurlyeq w$ holds true. What is left to prove is that

$$\neg(w \preccurlyeq y) \tag{3}$$

holds true as well. Let us assume the opposite. Then we gain with (1) that

$$w \preccurlyeq y \preccurlyeq z$$

holds true. With the transitivity of $\preccurlyeq$ then follows, that

$$w \preccurlyeq z.$$

That is a contradiction to (2) and hence $\prec$ has to be transitive as well.

■

During this thesis we are going to define various order relations which are based on a cone $K \subseteq \mathbb{R}^k$.

Definition 1.4
A nonempty subset K of $\mathbb{R}^k$ is called a cone *in $\mathbb{R}^k$ iff $y \in K$ and $\lambda \geq 0$ implies $\lambda y \in K$. Clearly, if K is a cone, then $0 \in K$. Such a cone $K \in \mathbb{R}^k$ is said to be:*

1. proper *iff $K \neq \{0\}$ and $K \neq \mathbb{R}^k$,*
2. convex *iff $y, z \in K$ implies $y + z \in K$,*
3. pointed *iff $-y \notin K$ for all $y \in K$, i.e., $K \cap (-K) = \{0\}$.*

The set of all closed, convex and proper cones K in $\mathbb{R}^k$ is denoted as

$$\mathcal{K}^k := \{K \subseteq \mathbb{R}^k \mid K \text{ is a closed, convex and proper cone in } \mathbb{R}^k\}.$$

We have to remark that these notations are not unique in literature, unfortunately there exist different kinds of definitions, when it comes to cones. Some authors use the definition above, where $K \subseteq \mathbb{R}^k$ being a cone implies that K contains the zero vector $0 \in \mathbb{R}^k$ ([26], [22]), some as well regard blunt cones, i.e. they choose $\lambda > 0$ in the definition above and allow K to be the empty set as well ([13], [33]).
Now we will present the first two binary relations induced by a cone.

Definition 1.5
The binary relations $\preceq_K$, $\prec_K$ on $\mathbb{R}^k$ induced by $K \in \mathcal{K}^k$, are defined as follows. For all $y, z \in \mathbb{R}^k$ it holds

$$\begin{aligned} y \preceq_K z &\quad :\Longleftrightarrow \quad z - y \in K, \\ y \prec_K z &\quad :\Longleftrightarrow \quad y \preceq_K z \text{ and } \neg(z \preceq_K y). \end{aligned}$$

The following corollary describes how the properties of a cone K influence the properties of the binary relations $\preceq_K$, $\prec_K$ induced by K.

Corollary 1.6
Let K be a cone in $\mathbb{R}^k$, if K is

1. *convex,*
   (i) $\preceq_K$ becomes a pre-order,

(ii) $\prec_K$ *becomes a strict partial order,*

2. *convex and pointed,*

   (i) $\preceq_K$ *becomes a partial order,*

   (ii) $y \prec_K z \iff y \preceq_K z$ *and* $y \neq z, \quad \forall y, z \in \mathbb{R}^k$.

Proof: 1. and 2. (i) follow directly by Theorem 1.20 of [13]. 1.(ii) follows by lemma 1.3. And 2.(ii) follows directly by the antisymmetry of $\preceq_K$. ■

If not denoted differently we will always regard closed, convex and proper cones $K \in \mathcal{K}^k$ throughout this thesis. The last definition of this subsection determines the conical hull of a subset of $\mathbb{R}^k$ and will be of importance for some of the proofs to follow.

Definition 1.7
Let Y *be a subset of* $\mathbb{R}^k$. *Then the* conical hull *of* Y *is defined to be*

$$cone(Y) := \Big\{ \sum_{i=1}^{l} \alpha_i y_i \;\Big|\; y_i \in Y,\, \alpha_i \in \mathbb{R},\, \alpha_i \geq 0,\, \forall i, l \in \mathbb{N} \Big\}.$$

1.3.2 Polyhedral Cones

Next we are going to define a very important type of cones for this thesis, the polyhedral cones in $\mathbb{R}^k$. To do so we will first need the following.

Definition 1.8
Let $l \in \mathbb{N}$ *and for all* $i \in \{1, 2, \ldots, l\}$, $a_i \in \mathbb{R}^k$ *be the* i*-th row vector of the matrix* $A \in \mathbb{R}^{l \times k}$, *then the* i*-th* half space *induced by* A *is given by*

$$H_i(A) := \{ y \in \mathbb{R}^k \mid \langle a_i, y \rangle \geq 0 \}$$

and the i*-th* hyperplane *induced by* A *is defined as*

$$P_i(A) := \{ y \in \mathbb{R}^k \mid \langle a_i, y \rangle = 0 \}.$$

Figure 1 illustrates the i-th hyperplane and the i-th half space $P_i(A)$ induced by some matrix $A \in \mathbb{R}^{l \times 3}$.

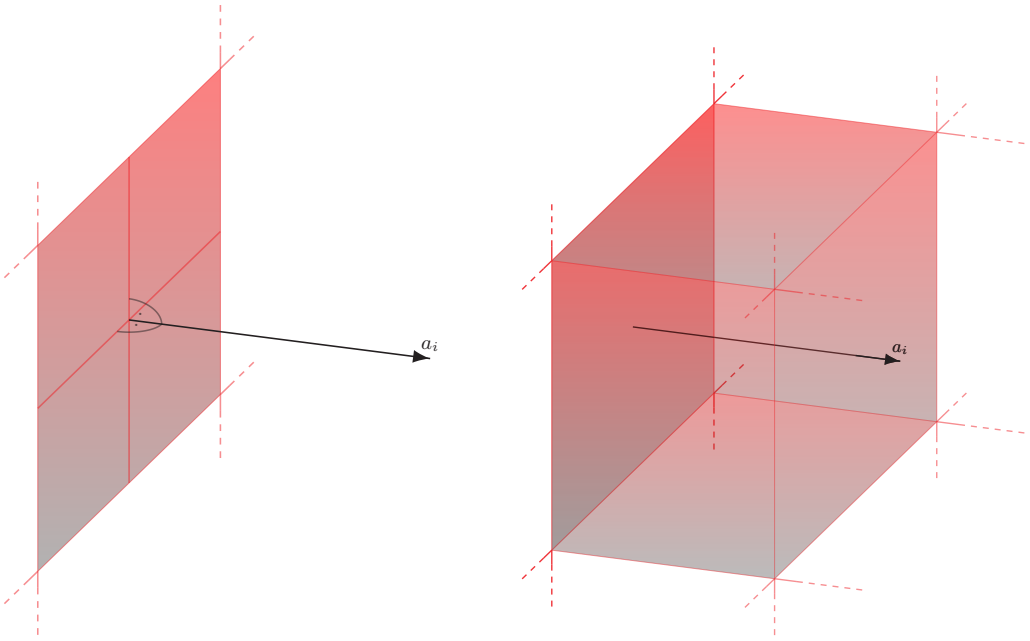


Figure 1: Illustration of the $P_i(A)$ (left) and $H_i(A)$ (right)

Now we are able to present the actual definition of a polyhedral cone.

Definition 1.9
Let $l \in \mathbb{N}$ and for all $i \in \{1, 2, \ldots, l\}$, $a_i \in \mathbb{R}^k$ be the i-th row vector of the matrix $A \in \mathbb{R}^{l \times k}$, then the polyhedral cone $K(A)$ *induced by the matrix A is defined as the intersection of all half spaces $H_i(A)$, $i \in \{1, 2, \ldots, l\}$ induced by A, i.e.*

$$K(A) := \bigcap_{i=1}^{l} H_i(A).$$

Notice that $K(A)$ is a closed convex cone for all $A \in \mathbb{R}^{l \times k}$ (see e.g. [13], [33]).

Now we confine ourselves even more onto a new definition of a polyhedral cone induced by a special kind of matrix, since the class of cones we gain by doing so, will play a main role in this thesis. In order to keep the reader's focus on our main goal of determining a new definition of robustness and keeping it from slipping off into the depths of polyhedral theory we have put those definitions and sentences on this topic which are not of main concern, into appendix A.

Definition 1.10
Let $A \in \mathbb{R}^{k \times k}$ be a square matrix with $rg(A) = k$ and row vectors $a_i, i \in \{1, 2, \ldots, k\}$. We denote the class of polyhedral cones $K(A)$ in $\mathbb{R}^k$ induced by

such a matrix $A \in \mathbb{R}^{k\times k}$ *as the class of* k-edged cones *in* $\mathbb{R}^k$. *The* i-th edge $E_i(A)$ *of* $K(A)$ *is given as*

$$E_i(A) := \left(\bigcap_{\substack{j=1,\\ j\neq i}}^{k} P_j(A) \right) \cap H_i(A).$$

The here stated definition of an edge of a k-edged cone does not intervene with the common definition of an edge of an arbitrary cone used in literature. This assertion is proven in lemma A.7 of appendix A. Furthermore the following corollary holds true.

Corollary 1.11
Let $K(A)$ *be a* k*-edged cone in* $\mathbb{R}^k$, *then it follows that* $K(A)$ *is pointed.*

Proof: Follows directly by theorem A.3 (ii) in Addendum A. ■

Definition and Example 1.12
Let $I^k \in \mathbb{R}^{k\times k}$ *be the* k*-dimensional identity matrix and let us define*

$$\mathbb{R}^k_+ := \{y \in \mathbb{R}^k \,|\, y_i \geq 0 \,\forall i \in \{1, 2, \ldots, k\}\}.$$

Notice that the k*-edged cone induced by* I^k *equals* $\mathbb{R}^k_+$, *i.e.* $K(I^k) = \mathbb{R}^k_+$.

The following example of a k-edged cone, which is our own creation, will accompany us throughout this thesis and will play an important role in the computational methods of section 2 and section 3 on cone robustness and uncertain optimization respectively.

Example 1
Let the matrix $A(\delta) \in \mathbb{R}^{k\times k}$ be defined as $A(\delta) := \big(a_{ij}(\delta)\big)_{i,j\in\{1,2,\ldots,k\}}$ with

$$a_{ij}(\delta) := \begin{cases} \frac{\tan(\delta)}{\sqrt{k-1}-(k-2)\tan(\delta)}, & \text{if } i \neq j, \\ 1, & \text{if } i = j. \end{cases} \tag{4}$$

Then $A(\delta)$ induces the k-edged cone $K(A(\delta)) \subset \mathbb{R}^k$, for all $\delta \in D := \left[0, \arctan\left(\frac{1}{\sqrt{k-1}}\right)\right)$, which has the edges

$$E_i(A(\delta)) = \left\{\lambda e^i(\delta) \,|\, \lambda \geq 0\right\}, \text{ with } e^i(\delta) := \begin{pmatrix} e^i_1(\delta) \\ e^i_2(\delta) \\ \vdots \\ e^i_k(\delta) \end{pmatrix} \tag{5}$$

$$\text{and } e^i_j(\delta) := \left\{ \begin{array}{ll} -\frac{\tan(\delta)}{\sqrt{k-1}} & \text{if } j \neq i, \\ 1 & \text{if } j = i \end{array} \right\}, \ \forall i \in \{1, 2, \ldots, k\}\,.$$

By the choice of D as a half opened and not closed interval we ensure that $K(A(\delta))$ stays pointed and avoid that the matrix $A(\delta)$ gets a rank smaller than k respectively, as presented in the proof of this example. In order to build some tension and keep the reader's interest on what ever might come next and what this dubious construction of a k-edged cone might be of use for, we do not give an explanation for the functionality of this particular definition of the cone $K(A(\delta))$ at this point of the thesis. But we present a graphical illustration of $K(A(0.2))$ as a small hint in figure 2.

Proof: The proof of this example is given in Addendum B. ■

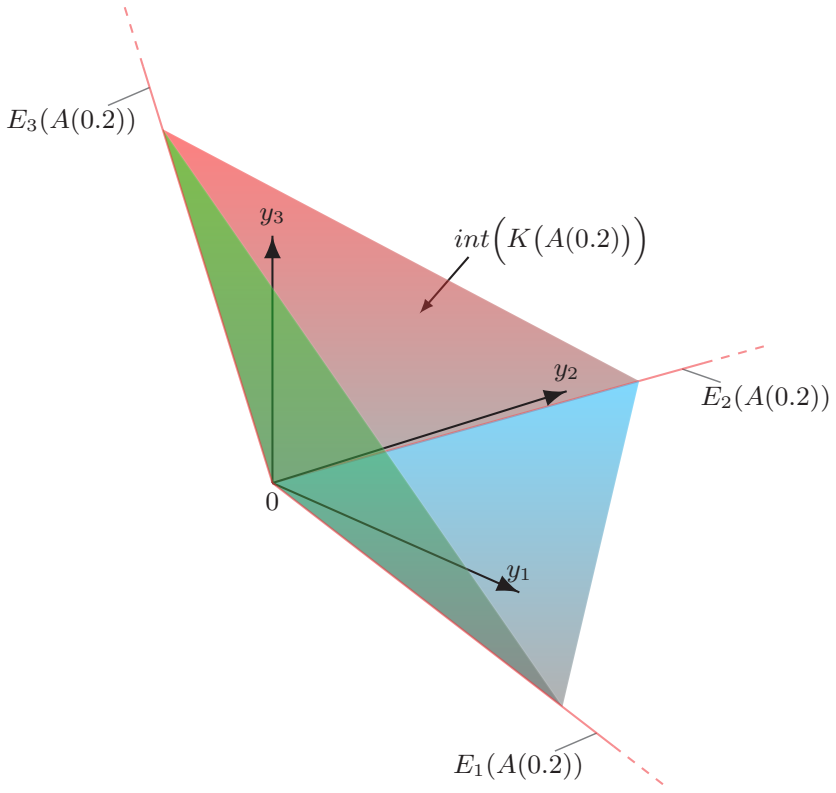


Figure 2: Illustration of the 3-edged cone $K(A(0.2))$ of example 1

The following theorem is a direct consequence of a main theorem in polyhedral theory and will be of importance for the continuation of example 1.

Theorem 1.13
Let $K(A)$ be a k-edged cone, then the following holds true,

$$K(A) = cone\Big(\big\{E_i(A) \,|\, i \in \{1, 2, \ldots, k\}\big\}\Big).$$

Proof: The statement is a direct consequence of theorem 2.15 in [28], with theorem 1.11 and definition 1.4. ■

Now we can move on to the notations that we use in this thesis on the topic of deterministic multiobjective optimization, i.e. multiobjective optimization without any perturbations included.

1.3.3 Deterministic Multiobjective Optimization

If not cited differently we are always regarding the following deterministic multiobjective optimization problem $\mathcal{P}_K$.

Definition 1.14
Let $K \in \mathcal{K}^k$ be a closed convex and proper cone, then the multiobjective optimization problem induced by K *is given by*

$$\mathcal{P}_K: \quad \min{}_K f(x) \quad s.t. \quad x \in \mathcal{X},$$

where the set $\mathcal{X} \subseteq \mathbb{R}^n$ contains all feasible solutions *and is called* decision space *or* feasible set*, $\mathbb{R}^n \backslash \mathcal{X}$ contains all infeasible solutions, $f : \mathcal{X} \to \mathbb{R}^k$, the function to be optimized is called the* objective map *and $Y := f(\mathcal{X}) = \{f(x) \in \mathbb{R}^k \mid x \in \mathcal{X}\}$ is the* objective space*.*

The notation $\min_K$ means here, that we are going to 'minimize' the objective function with respect to the ordering relation $\preceq_K$ induced by K, i.e. we are searching for efficient solutions w.r.t. K.

Definition 1.15 (Efficiency)
Let $\mathcal{P}_K$ be a multiobjective optimization problem induced by $K \in \mathcal{K}^k$. A solution $x^0 \in \mathcal{X}$ is called efficient w.r.t. K *iff*

$$f(x) \preceq_K f(x^0) \text{ for some } x \in \mathcal{X} \implies f(x^0) \preceq_K f(x).$$

The efficient set *of $\mathcal{P}_K$ is denoted by $\mathcal{S}_K := \{x \in \mathcal{X} \,|\, x \text{ is efficient w.r.t. } K\}$ and the* efficient front *of $\mathcal{P}_K$ then is defined to be $\mathcal{E}(\mathcal{P}_K) := f(\mathcal{S}_K)$.*

Unfortunately there does not exist a unique definition for efficiency in literature, some authors use Pareto optimal, nondominated or minimal for what we call efficient. Notice as well, that there exist other variants as the efficiency concept denoted here, such as *weak*, *strong* or *proper* efficiency [13], [3]. In general, though, one is mainly interested in finding efficient solutions and we will stick to this concept. Also there exist various equivalent alternatives of the formulation for the definition of an efficient solution given above. One of these we would like to point out to because of its popularity.

Corollary 1.16
A solution $x^0 \in \mathcal{X}$ of a multiobjective optimization problem $\mathcal{P}_K$ is efficient w.r.t K, iff

$$\text{there exists no } x \in \mathcal{X}, \text{ s.t. } f(x) \prec_K f(x^0). \tag{6}$$

Proof: Here the equivalence to definition 1.15 follows directly by the definition of our strict partial order $\prec_K$ in 1.5. ■

Next we need the definition of domination between two solutions of our optimization problem $\mathcal{P}_K$.

Definition 1.17 (Domination)
Let $\mathcal{P}_K$ be a multiobjective optimization problem induced by $K \in \mathcal{K}^k$.
If $f(\bar{x}) \prec_K f(x)$ holds true for two feasible solutions $\bar{x}, x \in \mathcal{X}, \bar{x} \neq x$, then the solution x is called dominated by $\bar{x}$ w.r.t. K *and synonymously it is said, that $\bar{x}$* dominates x w.r.t. K. *If for two arbitrary feasible solutions $x, \bar{x} \in \mathcal{X}, \bar{x} \neq x$ neither $f(x) \prec_K f(\bar{x})$ nor $f(\bar{x}) \prec_K f(x)$ is fulfilled, they are called* incomparable w.r.t. K *to one another.*

Notice that if $K = \mathbb{R}^k_+$ holds true, we gain the usual and well studied definitions of efficiency and domination initially introduced to applied sciences by Edgeworth (1881) [11] and Pareto (1906)[39]. An efficient solution w.r.t. $\mathbb{R}^k_+$, then also called *Edgeworth-Pareto optimal* solution, is a point $x^0 \in \mathcal{X}$ s.t. there exists no other $x \in \mathcal{X}$ with

$$f_i(x) \leq f_i(x^0) \quad \text{for all } i \in \{1, 2, \ldots, k\},$$

and

$$f_j(x) < f_j(x^0) \quad \text{for at least one } j \in \{1, 2, \ldots, k\}.$$

We will also call the cone $\mathbb{R}^k_+$ the *Edgeworth-Pareto cone* from here on.

The following corollary presents equivalent definitions to the definitions of efficiency and dominance for pointed and convex cones, which are more vivid and which will be of use for some of the proofs to follow.

Corollary 1.18
Let $x, \bar{x} \in \mathcal{X}$ be two feasible solutions of the multiobjective optimization problem $\mathcal{P}_K$ induced by a pointed cone $K \in \mathcal{K}^k$. Then

$$x \text{ dominates } \bar{x} \text{ w.r.t. } K \iff f(\bar{x}) \in \{f(x)\} + K \setminus \{0\}. \tag{7}$$

And

$$x^0 \text{ is efficient w.r.t. } K \iff$$
$$\{f(x^0)\} - K \setminus \{0\} \quad \text{does not contain any } f(x) \text{ with } x \in \mathcal{X}. \tag{8}$$

Proof: Equation (7) and (8) follow directly by corollary 1.6 point 2.(*ii*) and definition 1.17 or corollary 1.16 respectively. ■

The following simple example will illustrate us the above given definitions.

Example 2
We are regarding the multiobjective optimization problem $\mathcal{P}_K$ with objective map $f := id^2$, being the identity function in $\mathbb{R}^2$. The problem inducing cone K equals the Edgeworth-Pareto cone $\mathbb{R}^2_+$ and the feasible set is given by $\mathcal{X} := \{x \in \mathbb{R}^2 \mid (x_1 - 1)^2 + (x_2 - 1)^2 \leq 0\}$.
In figure 3, the objective space Y is depicted as the blue colored set, the red colored part of which is the efficient front $\mathcal{E}(\mathcal{P}_K)$ of $\mathcal{P}_K$. The solution $x^0 \in \mathcal{S}_K$ is efficient w.r.t. K and dominates the solution $\bar{x} \in \mathcal{X}$. On the left of figure 3, the inclusion $f(\bar{x}) \in \{f(x^0)\} + \mathbb{R}^2_+ \setminus \{0\}$ visualizes the domination of x^0 over $\bar{x}$ in the sense of equation (7) as well as the efficiency of x^0 in the sense of equation (8). On the right, $f(x^0) \in \{f(\bar{x})\} - \mathbb{R}^2_+ \setminus \{0\}$ visualizes the inefficiency of $\bar{x}$ in the sense of the negation of equation (8). The solution $\hat{x} \in \mathcal{X}$ neither is dominated by x^0 w.r.t. K nor does it dominate x w.r.t. K, thus those two solutions are incomparable w.r.t. K to each other.

In the literature on multiobjective optimization an ordering relation inducing cone K sometimes is denoted as *domination cone* [44] since, regarding equation (7) of corollary 1.18, one can interpret the cone $K \setminus \{0\}$ without 0 as the set of all *dominated directions* in $\mathbb{R}^k$. Thus the nonzero vectors in the domination cone can be thought of as 'bad' or 'dominated' directions to travel within our objective space and the choice of the cone K represents the preference attitude of the decision maker. Usually next to properness and convexity of our cone K pointedness is supposed as well. In other words, not all directions are allowed to be dominated directions, there has to exist at least one dominated direction, the sum of two dominated directions $k_1, k_2 \in K \setminus \{0\}$ is again a dominated direction, $k_1 + k_2 \in K \setminus \{0\}$ and that if $k \in K \setminus \{0\}$ is a dominated direction then the inverse direction is not allowed

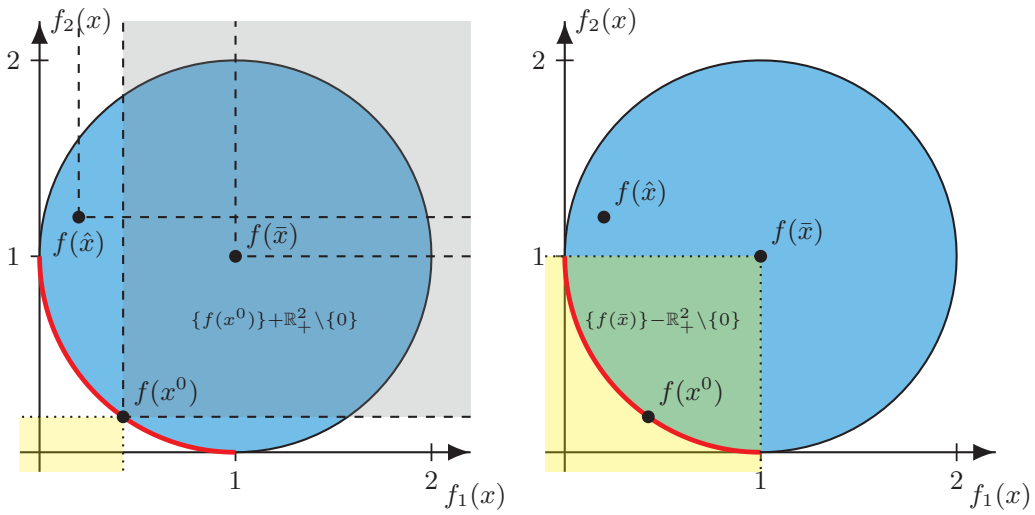


Figure 3: Illustration of equation (7), (8) (left) and ¬(8) (right) on example 2

to be a dominated direction as well, i.e. $-k \notin K \setminus \{0\}$. Therefor we will always regard closed convex proper and pointed cones K in $\mathbb{R}^k$ from here on, if not denoted differently.

2 Cone Robustness

Within this section, we are always regarding the multidimensional optimization problem $\mathcal{P}_K$ induced by a closed, convex, proper and pointed cone $K \in \mathcal{K}^k$ in $\mathbb{R}^k$ and let $\mathcal{C}_1, \mathcal{C}_2$ be two arbitrary nonempty subsets of $\mathcal{K}^k$, if not denoted differently.

As the literature review of section 1.2 has shown, there exist a lot of different possibilities where perturbations within an optimization process may occur and at least as many ways to handle these uncertainties to gain some kind of robust efficient solution. The concept of cone robustness discusses perturbations within the cone K which defines our ordering relation on the objective space. As discussed in section 1.3.3, this ordering relation can be interpreted as the decision makers preferences which also reflect his attitude towards risk. Such perturbations may occur in real world optimization problems if the decision makers change over the period of optimization. In Australia, for instance, military decision makers who are involved in long-term planning of capability development projects typically rotate into different positions ev-

ery two years, although most of such projects span at least five to ten years [1]. Uncertainties within the ordering relation implying cone also are of importance, if not the decision maker changes in person, but his preferences vary over time. An example for perturbations within the decision maker's preferences is given in portfolio optimization. Regarding a real estate broker or a portfolio analyst, preferences might change over a longer period of time or even on a daily basis in accordance to the developments of the real estate or stock market respectively.

Hence we are now searching for solutions which are robust with respect to perturbations in our cone K. More precisely, given a finite family $\mathcal{C} \subset \mathcal{K}^k$ of closed, proper, convex and pointed cones containing our initial cone $K \in \mathcal{C}$, we are looking for the solution which is efficient with respect to every cone C in $\mathcal{C}$. Since the union of such a family of cones is a cone as well, we are looking for solutions which are efficient w.r.t. the *union cone* $U := \bigcup_{C \in \mathcal{C}} C$. The following, simple to prove, theorem is of upmost importance for our concept of cone robustness.

Theorem 2.1
Let K_1 and K_2 be two pointed cones in $\mathcal{K}^k$, with K_1 being a true subset of K_2, i.e.

$$K_1 \subset K_2$$

and $\mathcal{P}_{K_1}, \mathcal{P}_{K_2}$ the multiobjective optimization problems induced by K_1 and K_2 respectively, with identical objective map $f : \mathcal{X} \to \mathbb{R}^k$ and decision space $\mathcal{X}$. Then it follows that the efficient front of $\mathcal{P}_{K_2}$ is a subset of the efficient front of $\mathcal{P}_{K_1}$, i.e.

$$\mathcal{E}(\mathcal{P}_{K_2}) \subseteq \mathcal{E}(\mathcal{P}_{K_1}).$$

Proof: Let $y^0 \in \mathcal{E}(\mathcal{P}_{K_2})$ be an element of the efficient front of $\mathcal{P}_{K_2}$, that is equivalent to the existence of an efficient solution $x^0 \in \mathcal{S}_{K_2}$ w.r.t. K_2 for which $f(x^0) = y^0$ holds true. With corollary 1.16, that again is equivalent to

$$\begin{aligned}
&\nexists x \in \mathcal{X} : \quad f(x) \prec_{K_2} f(x^0) && \overset{Cor.1.18}{\Longleftrightarrow} \\
&\nexists x \in \mathcal{X} : \quad f(x^0) \in f(x) + K_2 \setminus \{0\} && \overset{K_1 \subset K_2}{\Longleftrightarrow} \\
&\nexists x \in \mathcal{X} : \quad f(x^0) \in f(x) + K_1 \setminus \{0\} && \overset{Cor.1.18}{\Longleftrightarrow} \\
&\nexists x \in \mathcal{X} : \quad f(x) \prec_{K_1} f(x^0)
\end{aligned}$$

which is equivalent to $x^0 \in \mathcal{S}_{K_1}$ being efficient w.r.t. K_1, with corollary 1.16. And at last, this is equivalent to $y^0 \in \mathcal{E}(\mathcal{P}_{K_1})$ being an element of the efficient front of $\mathcal{P}_{K_1}$. ■

This result means for our aim of finding robust solutions with respect to perturbations in the ordering relation inducing cone K, that the 'bigger' the perturbations of K, the larger our union cone U will get and the smaller the set of robust solutions will become. To model such perturbations within K we will need the next definition.

Definition 2.2 (Ascending Cone Mapping)
Let $D \subseteq \mathbb{R}$ with $0 \in D$, let $\mathcal{C}_1, \mathcal{C}_2$ be arbitrary nonempty subsets of $\mathcal{K}^k$, then an injective mapping

$$\mathrm{c} : \mathcal{C}_1 \times D \to \mathcal{C}_2,\ (K, \delta) \mapsto \mathrm{c}(K, \delta)$$

is called an ascending cone mapping *iff the following holds true:*

(i) $\mathrm{c}(K, 0) = K$, *for all* $K \in \mathcal{C}_1$,

(ii) for all $K \in \mathcal{C}_1$ *and* $\delta_1, \delta_2 \in D$ *with* $\delta_1 < \delta_2$ *follows that* $\mathrm{c}(K, \delta_1) \subset \mathrm{c}(K, \delta_2)$.

Given an element $\mathrm{c}(K, \delta) \in \mathcal{C}_2$ *of the image set of* c, *the corresponding element* $\delta \in D$ *of the domain is then called the* perturbation factor *of* $K \in \mathcal{C}_1$. *Such an ascending cone mapping* c *is called* proper *iff* $D \subset \mathbb{R}_+$ *is a bounded non-negative subset of* $\mathbb{R}$ *and the subsets* $\mathcal{C}_1, \mathcal{C}_2 \subset \mathcal{K}^k$ *do only contain pointed cones.*

Thus an ascending cone mapping models the perturbation within K in the sense that its image, which is the cone representing our union cone U, expands in dependence of its perturbation factor δ. The following example is a continuation of example 1 and may cast some light on the mysteries of the definition of a k-edged cone presented there.

Example 3
Let $\mathcal{C}_1 := \{\mathbb{R}^k_+\}$, $D := \left[0, \arctan\left(\frac{1}{\sqrt{k-1}}\right)\right)$ and let the matrix $A(\delta) \in \mathbb{R}^{k \times k}$ be defined as in example 1. Then the mapping

$$\bar{\mathrm{c}} : \mathcal{C}_1 \times D \to \mathcal{C}_2,\ (K, \delta) \mapsto K(A(\delta)),$$

is a proper ascending cone mapping, which maps $(\mathbb{R}^k_+, \delta)$ onto the pointed k-edged cone

$$K(A(\delta)) = cone\Big(\big\{E_i(A(\delta)) \,|\, i \in \{1, 2, \ldots, k\}\big\}\Big), \tag{9}$$

for each $\delta \in D$.

Proof: A proof of example 3 is given in appendix C ■

An instant consequence of theorem 2.1 and definition 2.2 is the following corollary, which will be of importance for the algorithm presented in the next subsection.

Corollary 2.3
Let $\mathrm{c} : \mathcal{C}_1 \times D \to \mathcal{C}_2$ *be a proper ascending cone mapping defined as in definition 2.2, let* $K \in \mathcal{C}_1$ *and* $(\delta_i)_{i\in\mathbb{N}}$ *be a strictly monotonically increasing sequence in* D *then the sequence of multiobjective optimization problems* $(\mathcal{P}_{\mathrm{c}(K,\delta_i)})_{i\in\mathbb{N}}$ *with identical objective maps and feasible sets results in a descending sequence of efficient fronts* $(\mathcal{E}(\mathcal{P}_{\mathrm{c}(K,\delta_i)}))_{i\in\mathbb{N}}$, *i.e.*

$$\mathcal{E}(\mathcal{P}_{\mathrm{c}(K,\delta_i)}) \supseteq \mathcal{E}(\mathcal{P}_{\mathrm{c}(K,\delta_{i+1})}) \quad \textit{for all } i \in \mathbb{N}.$$

Proof: Follows directly by theorem 2.1 and definition 2.2. ■

Now we can present our new definition of robustness.

Definition 2.4 (Cone robust efficiency)
Let $\mathcal{P}_K$ *be a multiobjective optimization problem induced by a cone* $K \in \mathcal{C}_1$ *let* $\mathrm{c} : \mathcal{C}_1 \times D \to \mathcal{C}_2$ *be a proper ascending cone mapping and let* $\delta \in D$ *hold true. A solution* $x^0 \in \mathcal{X}$ *of* $\mathcal{P}_K$ *then is called a* cone robust efficient solution w.r.t. *the pair* (c, δ) *iff* x^0 *is an efficient solution w.r.t.* $\mathrm{c}(K, \delta)$ *of* $\mathcal{P}_{\mathrm{c}(K,\delta)}$, *i.e. if* $x^0 \in \mathcal{S}_{\mathrm{c}(K,\delta)}$, *where* $\mathcal{S}_{\mathrm{c}(K,\delta)}$ *is called the* cone robust efficient set w.r.t. (c, δ) *of* $\mathcal{P}_K$.

Given an ascending cone mapping c we also define a mapping which assigns a maximal degree of resistance capability against perturbations within the ordering relation inducing cone K to efficient solutions.

Definition 2.5 (Cone robustness degree)
Let $\mathcal{P}_K$ *be a multiobjective optimization problem induced by the cone* $K \in \mathcal{C}_1$, *let* $\mathrm{c} : \mathcal{C}_1 \times D \to \mathcal{C}_2$ *be a proper ascending cone mapping, let* $\mathcal{F} \subseteq \mathcal{S}_K$ *be a set of efficient solutions of* $\mathcal{P}_K$ *and let* $x^0 \in \mathcal{F}$. *Then is the* cone robustness degree of x^0 *with respect to the pair* $(\mathcal{F}, \mathrm{c})$ *given by*

$$CRD^{\mathrm{c}}_{\mathcal{F}}(x^0) \quad := \quad \sup\{\delta \in D \mid \nexists x \in \mathcal{F} : \ f(x) \prec_{\mathrm{c}(K,\delta)} f(x^0)\}.$$

We gave this new robustness measure the name cone robustness degree since, to put it crudely, it yields for each efficient solution $x^0 \in \mathcal{F} \subseteq \mathcal{S}_K$ out of a set of efficient solutions a degree one can perturb the cone K w.r.t c, s.t. this solution does not become dominated. The following examples continue example 1 and 3 and will illustrate this last definition on the basis of the statements presented in corollary 1.18.

Example 4
Given the proper ascending cone mapping $\mathrm{c} : \mathcal{C}_1 \times D \to \mathcal{C}_2$, let $\mathcal{P}_K$ be a multiobjective optimization problem induced by a cone $K \in \mathcal{C}_1$, let $x^0, \bar{x} \in S_K$ and let $\delta \in D$ hold true. Now the cone robustness degree of x^0 w.r.t. (c, S_K) is a degree one can perturb the cone K w.r.t c, s.t. this solution stays efficient in the sense of equation (8) of corollary 1.18 or as long as it stays undominated by another solution $\bar{x} \in \mathcal{X}$ in the sense of equation (7) of corollary 1.18 respectively. Figure 4 gives us a visual illustration of the cone robustness degree with respect to the ascending cone mapping $\bar{\mathrm{c}}$ of example 3 for a two-dimensional objective space. As can be seen here, our ascending cone mapping $\bar{\mathrm{c}}$ is chosen in such a manner, that it yields for each Edgeworth-Pareto efficient solution a cone robustness degree which displays the actual angle, we are tilting the edges of the Edgeworth-Pareto cone. In the following example we will prove this assertion.

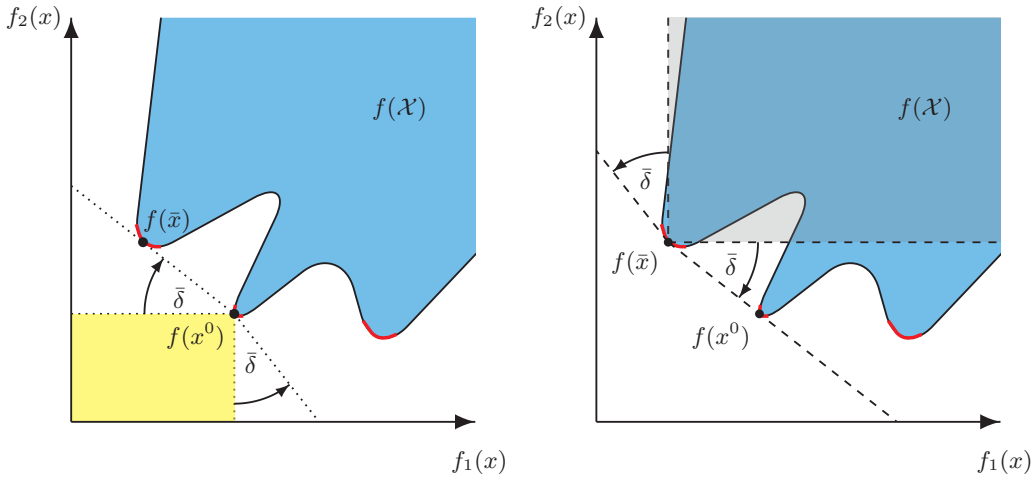


Figure 4: Illustration of $CRD^{\bar{\mathrm{c}}}_{S_K}(x^0) = \bar{\delta}$ in the sense of (8) (left) and (7) (right)

Example 5
Let the matrix $A(\delta) \in \mathbb{R}^{k \times k}$ be defined as in example 1 and the ascending cone mapping $\bar{\mathrm{c}} : \mathcal{C}_1 \times D \to \mathcal{C}_2$ as in example 3. Let $e_i(\delta)$ be the i-th edge of the k-edged cone $K(A(\delta))$, defined as in example 1 for all $\delta \in D$ and $i \in \{1, 2, \ldots, k\}$. Then the vectors $e^i(0)$ or $e^i(\delta)$ are positively spanning the i-th edge of the polyhedral cone $\bar{\mathrm{c}}(\mathbb{R}^k_+, 0) = \mathbb{R}^k_+$ or $\bar{\mathrm{c}}(\mathbb{R}^k_+, \delta) = K(A(\delta))$ respectively, for $\delta \in D, i \in \{1, 2, \ldots, k\}$, as proven in example 1. Those vectors have an included angle of δ and lie on the plane spanned by the i-th

unit vector i_i and the vector $v_i := \sum_{\substack{j=1,\\ j\neq i}}^{k} i_j$ containing ones in each objective except in the i-th objective stands a 0. I.e. $e^i(0), e^i(\delta) \in lin(\{i_i, v_i\})$ for all $\delta \in D$ and $i \in \{1, 2, \ldots, k\}$.

Figure 5 and 6 depict how the edges of the cone $\mathbb{R}^3_+$ tilt in the three-dimensional case for our example. Figure 5 shows the Edgeworth-Pareto cone $\mathbb{R}^3_+$ and figure 6 the opened cone $\bar{c}(\mathbb{R}^3_+, \delta)$ with perturbation factor $\delta = 0.3$. In figure 5 the yellow snippet of a plane visualizes the plane spanned by i_3 and v_3 and the red, green or blue snippets depict the intersection of $\bar{c}(\mathbb{R}^3_+, 0.3)$ with the hyperplane induced by the first, second or third row vector of $A(0.3)$ respectively, which are exactly the so called facets whose union is the boundary of the 3-edged cone $K(A(0.3))$ (see e.g. [27]).

Proof: Since for all $\delta \in D$ and $i \in \{1, 2, \ldots, k\}$,

$$e^i(0) = i_i \qquad \text{and} \qquad e^i(\delta) = i_i - \frac{tan(\delta)}{\sqrt{k-1}} v_i$$

holds true, we gain $e^i(0), e^i(\delta) \in lin(\{i_i, v_i\})$ for all $\delta \in D$ and $i \in \{1, 2, \ldots, k\}$. To prove that the included angle of the vectors $e^i(0)$ and $e^i(\delta)$ equals δ, we use the following property of the scalar product. For two vectors $u, v \in \mathbb{R}^n$,

$$\langle u, v \rangle = \cos(\phi) \|u\| \|v\| \tag{10}$$

holds true, with ϕ being the included angle of u and v (see e.g. [25]). The following equation is true for all $\delta \in \mathbb{R}$ (see e.g. [16])).

$$\begin{aligned}
& \cos^2(\delta) + \sin^2(\delta) = 1 \\
\overset{tan=\frac{sin}{cos}}{\iff} \quad & (1 + \tan^2(\delta)) \cos^2(\delta) = 1 \\
\iff \quad & \cos(\delta) = \frac{1}{\sqrt{1 + \tan^2(\delta)}} \\
\overset{e^i(0)=i_i,(5)}{\iff} \quad & \cos(\delta) = \frac{\langle e^i(0), e^i(\delta) \rangle}{\|e^i(0)\| \|e^i(\delta)\|}.
\end{aligned}$$

Thus with (10) follows the assertion. ■

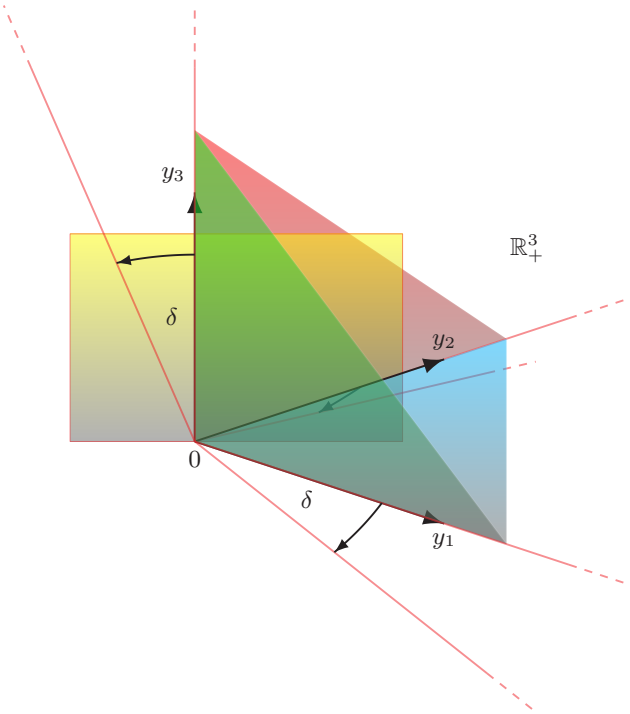


Figure 5: Illustration of the opening process of $\mathbb{R}^3_+$ caused by $\bar{c}$.

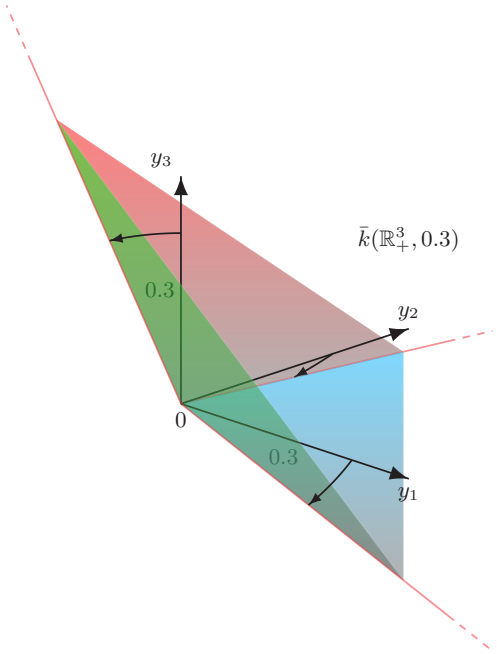


Figure 6: Opened cone $\bar{c}(\mathbb{R}^3_+, 0.3)$.

In the following subsection we will apply the ascending cone mapping $\bar{c}$ of the examples above within an evolutionary algorithm. This algorithm is developed to evolve a well spread population of solutions which are led towards a predefined cone robust efficient part of the efficient front, with an a posteriori assignment of the actual cone robustness degree.

2.1 Computational Methods

Within this section, we confine ourselves onto the following class of multiobjective optimization problems induced by a closed, convex, proper and pointed cone K in $\mathbb{R}^k$.

$$\bar{\mathcal{P}}_K: \quad \min_K f(x) \quad \text{s.t.} \quad x \in \mathcal{X},$$

with objective map $f : \mathbb{R}^n \to \mathbb{R}^k$, $l \in \mathbb{N}$ and feasible set $\mathcal{X} := \{x \in \mathbb{R}^n \,|\, g_i(x) \leq 0, \forall i \in \{1, 2, \ldots, l\}\}$, with inequality constraint functions $g_i : \mathbb{R}^n \to \mathbb{R}$ for all $i \in \{1, 2, \ldots, l\}$.

The restriction of the feasible set onto sets defined through $l \in \mathbb{N}$ inequality constraints is necessary since the algorithms presented in this section are working with a so called constraint violation comparator, described in detail in section 2.1.1. Moreover the confinement of the objective function f onto functions with domain $\mathbb{R}^n$ is mathematically necessary due to the usage of the crowding distance assignment within our algorithms, e.g. presented in [2]. We will give a more detailed explanation for those two required restrictions when the triggering tools, i.e. the constraint violation comparator and the crowding distance assignment, are introduced in detail in section 2.1.1.

In the introduction part of this thesis we have already presented the advantages evolutionary algorithms bring along in comparison with most of the conventional methods of solving multiobjective optimization problems. In this subsection we will use the results from above to develop an evolutionary algorithm searching for cone robust efficient solutions based on the non-dominated sorting based genetic algorithm II, `NSGA-II` (see e.g. [2]), whose procedure is presented in figure 7.
Here P_t denotes the parent population, O_t the offspring population and R_t their union of the t-th generation. For $i \in \mathbb{N}$, $\mathcal{F}_i$ denotes the i-th non-dominated front.

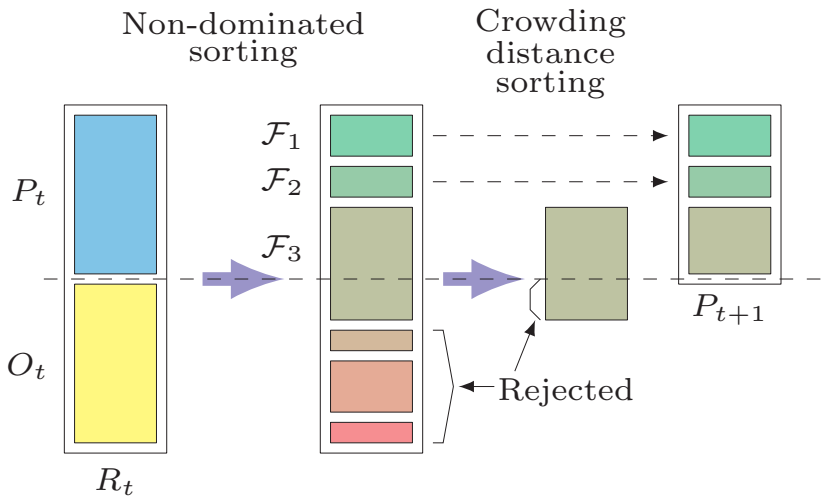


Figure 7: Procedure of `NSGA-II`, [2].

Non-dominated sorting is a highly efficient ranking scheme which is very popular in the field of population-based algorithms [40]. Given a multiobjective optimization problem $\mathcal{P}_K$ with feasible set $\mathcal{X} \subseteq \mathbb{R}^n$, the non-dominated sorting algorithm assigns to each solution $x \in P$ of a given set of solutions $P \subseteq \mathcal{X}$, which as well is called a *population* of solutions, a rank based on the number of solutions $\bar{x} \in P \setminus \{x\}$ dominating the given solution x. This yields a set of disjunct subsets $\{\mathcal{F}_i \subseteq P \,|\, \mathcal{F}_i \cap \mathcal{F}_j = \emptyset, \forall i \neq j, i, j \in \{1, 2, \ldots, l\}\}$ of P, for some $l \in \mathbb{N}$, where $\mathcal{F}_i$ contains exactly those solutions with rank i. I.e. $x \in P$ is in $\mathcal{F}_i$ iff x and $\hat{x}$ are incomparable w.r.t. K to each other, for all $\hat{x} \in \mathcal{F}_i$ and there exist exactly i solutions $\bar{x} \in P$ with $f(\bar{x}) \prec_K f(x)$. In accepted usage non-dominated sorting works with the usual Edgeworth-Pareto cone, for our purposes though, we will be using some arbitrary closed, convex, proper and pointed cone $K \in \mathcal{K}^k$.
Crowding distance sorting ranks the underlying solutions in accordance to their distribution in the objective space, therefor to each solution a value is assigned to, the so called crowding distance. This approach prevents the algorithm from getting stuck in local minima and guarantees a well distributed approximation of the efficient front as outcome of the `NSGA-II`. The procedure of the crowding distance assignment is explained in the next section in more detail.

2.1.1 Cone Robustness Based Evolutionary Algorithm (CREA)

Given a multiobjective optimization problem $\bar{\mathcal{P}}_K$ suffering under perturbations within K, represented by a proper ascending cone mapping $\mathrm{c} : \mathcal{C}_1 \times D \to \mathcal{C}_2$, the cone robustness based evolutionary algorithm CREA has two main tasks. First, evolve a population of well spread solutions which cover a region of the undisturbed efficient set $\mathcal{S}_K$ w.r.t. K that is cone robust efficient w.r.t. (c, δ) for some predefined perturbation factor $\delta \in D$. Secondly, assigning each so found solution its cone robustness degree. For the first step the algorithm uses the two parameters $\delta \in D$ and $\tau \in [0, 1]$, where δ is the perturbation factor of our cone K defining the area towards which our population evolves to and τ is a threshold ensuring a proper diversity of the population, as presented in detail later in algorithm 1. For the second step we have to choose the parameter $\sigma \in \mathbb{R}_+$ defining the step size of a proper monotonically increasing sequence of perturbation factors $(\delta_i)_{i \in \mathbb{N}}$ in order to assign a cone robustness degree to each of the found solutions. Algorithm 3 presents this process in detail and also explains the terminology of step size and properness.

In the following we assume a population size of $N \in \mathbb{N}$ for our *parent* and *offspring* populations, denoted as P_t and O_t, respectively, i.e. $|P_t| = N = |O_t|$. Let $R_t := P_t \cup O_t$ be their union, at any *generation* $t \in \mathbb{N}$, such as let $e \in \mathbb{N}$ be a counter for the current amount of evaluations of the objective function f of $\bar{\mathcal{P}}_K$ and $M \in \mathbb{N}$ the maximal amount of such evaluations, defining the limit of generations CREA is running for.

Algorithm 1 presents a framework for the creation of a new parent population within CREA. This procedure is oriented on the Proper Knee Based Evolutionary Algorithm (PKEA) presented by P. K. Shukla, M. A. Braun and H. Schmeck in [36] in 2013. The PKEA is developed to find proper knee regions, a part of the efficient front with good properties, more precisely which contains solutions below a user-defined threshold metric. Algorithm 1 orientates on the procedure used in PKEA to generate a new parent population while adjusting the level of diversity as well as the level of attraction towards the knee or robust region respectively via the user-defined threshold $\tau \in [0, 1]$.

Algorithm 1 : Creation of New Parent Population in `CREA`

Input: Finite set R of at least N solutions, proper ascending cone mappingc : $\mathcal{C}_1 \times D \to \mathcal{C}_2$, original cone $K \in \mathcal{C}_1$, perturbation factor $\delta \in D$ and threshold $\tau \in [0, 1]$.

1 **begin**

2 - Perform a non-dominated sorting to R_t w.r.t. the original cone K to identify the different fronts $\mathcal{F}_i, i = 1, 2, \ldots, l,\ l \in \mathbb{N}$.

3 - Set $j = 1$ and $P = \emptyset$.

4 **while** $|P \cup \mathcal{F}_j| < N$ **do**

5 - Assign crowding distance to all elements of $\mathcal{F}_j$.

6 - Replace P by $P \cup \mathcal{F}_j$.

7 - Replace j by $j + 1$.

8 **end**

9 - Perform a non-dominated sorting to $\mathcal{F}_j$ w.r.t. the perturbed cone $\mathrm{c}(K, \delta)$ to identify the different fronts $F_i^c, i = 1, 2, \ldots, m,\ m \in \mathbb{N}$.

10 - Add $\min\left\{\left\lceil \tau(N - |P|)\right\rceil, |F_1^c|\right\}$ arbitrary solutions of F_1^c to P and remove them of $\mathcal{F}_j$.

11 - Assign crowding distance to all elements of $\mathcal{F}_j$.

12 - Add $\left|N - |P|\right|$ solutions of $\mathcal{F}_j$ to P considering their crowding distance values.

13 **end**

Output: New parent set P.

Here the function $\lceil\cdot\rceil$ used in line 10 of the algorithm is the common ceil function $\lceil x\rceil := \min\{n \in \mathbb{Z} \mid n \geq x\}$, returning the smallest following integer for each real number. The crowding distances are already assigned in line 5, due to the fact that some of the selection mechanisms allowed in `CREA` work with crowding distance values as well. Note that the threshold τ controls how strong the population is driven towards the cone robust efficient set w.r.t. (c, δ) as well as it controls the diversity within this population. Especially a choice of the extremal values for τ will reflect this statement. I.e. a choice of $\tau = 0$ leads towards a search of the complete efficient front $\mathcal{E}(\bar{\mathcal{P}}_K)$ as the minimum in line 10 of algorithm 1 will become zero. A choice of $\tau = 1$ will result in a low diversity of the found population since the algorithm runs without any or little consideration of the crowding distance as $\left|N - |P|\right|$ in line 12 will become zero, if the cardinality $|F_1^c|$ of the first non-dominated set

w.r.t. $\mathrm{c}(K,\delta)$ is at least $\left|N-|P|\right|$ in line 10. A choice of τ within between those two extremal values on the other hand, e.g. $\tau = 0.5$, should result in a well spread population driven towards the part of the efficient set $\mathcal{S}_K$, which is cone robust efficient w.r.t. (c,δ). In order to assign the crowding distance to each element of $\mathcal{F}_j$ in line 5 and 11 of algorithm 1 we use the *crowding distance assignment* of the `NSGA-II`, whose framework is depicted in algorithm 2 and which is described more in more detail in [2].

Algorithm 2 : Crowding Distance Assignment of [2]

	Input: Set $\mathcal{F}$ of l solutions	
1	**begin**	
2	- **for each** i, set $\mathcal{F}[i]_{distance} = 0$	initialize distance
3	- **for each** objective m	
4	$\mathcal{F} = \mathrm{sort}(\mathcal{F}, m)$	sort after objective value
5	$\mathcal{F}[1]_{distance} = \mathcal{F}[l]_{distance} = \infty$	so that boundary points are always selected
6	for $i = 2$ to $(l-1)$	for all other points
7	$\mathcal{F}[i]_{dist} = \mathcal{F}[i]_{dist} + \frac{(\mathcal{F}[i+1].m_{dist} - \mathcal{F}[i+1].m_{dist})}{f_m^{max} - f_m^{min}}$	
8	**end**	

Here, $\mathcal{F}[i].m$ refers to the m-th objective map value of the i-th individual in the set $\mathcal{F}$ and the parameters f_m^{max} and f_m^{min} are the maximum and minimum values of the m-th objective map. For a more detailed explanation, we refer to its source [2].

Algorithm 3 shows a framework for the assignment process of the cone robustness degree used in `CREA`.

Algorithm 3 : Assignment of the Cone Robustness Degree in `CREA`

Input: Solution set $\mathcal{F}_1$, proper ascending cone mapping $\mathrm{c} : \mathcal{C}_1 \times D \to \mathcal{C}_2$ original cone $K \in \mathcal{C}_1$ and step size $\sigma \in \mathbb{R}_+$.

1 **begin**
2 - Set $i, i_l, l = 1$, $\delta_i = 0$, $\mathcal{F}_i^* = \mathcal{F}_1$, $\mathcal{F}_{i_l} = \mathcal{F}_i^*$ and $L_0 = \emptyset$.
3 **while** $\delta_i + \sigma < \sup\{\delta \,|\, \delta \in D\}$ **do**
4 **while** $|\mathcal{F}_i^*| = |\mathcal{F}_{i_l}|$ **and** $\delta_i + \sigma < \sup\{\delta \,|\, \delta \in D\}$ **do**
5 - Replace δ_i by $\delta_i + \sigma$.
6 - Perform a non-dominated sorting to $\mathcal{F}_{i_l}$ w.r.t. the perturbed cone $\mathrm{c}(K, \delta_i)$ to identify the different fronts $F_j^c, j = 1, 2, \ldots, p$, $p \in \mathbb{N}$.
7 - Replace i by $i + 1$ and $\mathcal{F}_i^*$ by F_1^c.
8 **end**
9 - Replace l by $l + 1$, i_l by i and $\mathcal{F}_{i_l}$ by $\mathcal{F}_i^*$.
10 - Assign cone robustness degree δ_{i-1} to all solutions in the fronts F_j^c, for all $j \in \{2, 3, \ldots, p\}$ and set $L_{l-1} = \mathcal{F}_{i_{l-1}} \setminus \mathcal{F}_{i_l} = \bigcup_{j=2}^{p} F_j^c$.
11 **end**
12 - Assign cone robustness degree δ_i to all solutions in F_1^c.
13 **end**

Note that within the assignment process of the cone robustness degree of algorithm 3 the monotonically increasing sequence $(\delta_i)_{i \in \mathbb{N}} \subset \mathbb{R}$ with step size σ is generated in line 2, 5 and 7, more precisely $\delta_{i+1} = \delta_i + \sigma$. This sequence is proper in the sense of being a monotonically increasing non-negative sequence of perturbation factors $(\delta_i)_{i \in \mathbb{N}} \subset \mathbb{R}_+$, with $\delta_1 = 0$. Simultaneously the sequence of solution sets $(\mathcal{F}_i^*)_{i \in \mathbb{N}} \subseteq \mathcal{F}_1$ is generated in line 2 and 7. This sequence contains the non-dominated solutions w.r.t. $\mathrm{c}(K, \delta_i)$ of $\mathcal{F}_{i_l}$ in each iteration $i \in \mathbb{N}$ of the inner loop. The subsequence $(\mathcal{F}_{i_l})_{l \in \mathbb{N}} \subseteq \mathcal{F}_1$ of $(\mathcal{F}_i^*)_{i \in \mathbb{N}}$ generated in line 2 and 9 is a strictly decreasing sequence of solution sets, i.e. $\mathcal{F}_{i_{l+1}} \subset \mathcal{F}_{i_l}$ for each iteration $l \in \mathbb{N}$ of the outer loop. Notice, that the sequence $(\mathcal{F}_{i_l})_{l \in \mathbb{N}}$ is decreasing is a direct consequence of the first stopping criterion of the inner loop and corollary 2.3 of section 2, when the sequence of multiobjective optimization problems regarded there is given through

$$\hat{\mathcal{P}}_{\mathrm{c}(K,\delta_i)} : \quad \min{}_{\mathrm{c}(K,\delta_i)} id^k \;\; s.t. \;\; x \in \mathcal{F}_{i_l},$$

for $i, l \in \mathbb{N}$. In line 9 of algorithm 3 we are removing the already dominated

solutions $\bigcup_{j=2}^{p} F_j^c$ out of the set of currently regarded solutions $\mathcal{F}_{i_l}$ in order to save the computational effort which would be necessary to perform a non-dominated sorting on the larger set $\mathcal{F}_1$. That this removal of already dominated solutions out of the currently regarded set does not have any impact on the cone robustness degree of a still non-dominated solution is proven in appendix D.
Thus, given a set of efficient solutions $\mathcal{F}_1$, algorithm 3 assigns the best approximation of its actual cone robustness degree $CRD^{\mathrm{c}}_{\mathcal{F}_1}(x^0)$ possible under the given step size to each obtained efficient solution $x^0 \in \mathcal{F}_1$ of $\bar{\mathcal{P}}_K$ w.r.t. K , i.e.

$$0 \leq CRD^{\mathrm{c}}_{\mathcal{F}_1}(x^0) - \bar{\delta} < \sigma, \tag{11}$$

where $\bar{\delta}$ is the cone robustness degree assigned to x^0 in line 10. Therefor does the assigned cone robustness degree $\bar{\delta}$ of x^0 converge towards the true cone robustness degree $CRD^{\mathrm{c}}_{\mathcal{F}_1}(x^0)$ as the step size σ converges towards 0. Note that none of the above mentioned sequences are infinite, due to the fact that we have a proper ascending cone mapping with restricted domain. Hence it would be correct to change the domains $\mathbb{N}$ of the sequences to restricted ones $\{1, 2, \ldots, \hat{m}\}, \hat{m} \in \mathbb{N}$ as well. We did not do this in the interest of clarity and we will leave it to the interested reader to assign the appropriate maxima $\hat{m}$, since these maxima are not of any importance for our above presented convergence deliberations. From here on we will refer to the set of solutions L_l as the *cone robustness level* l, due to the fact that it contains the solutions to whom algorithm 3 assigns a cone robustness degree in line 10 in the l-th iteration of the outer loop. Note that the higher the cone robustness level $l \in \mathbb{N}$ a solution $x^0 \in L_l$ is in, the higher is its assigned cone robustness degree. In order to allow the decision maker to plot each cone robustness level separately or as a whole together with an assigned legend, algorithm 3 as well creates external data in line 10 and 12 storing the assigned cone robustness degree and objective function values of all solutions within the current level, for each iteration of the outer loop.

Now we are able to present the complete framework of `CREA` in algorithm 4.

Algorithm 4 : `CREA`

Input: Multiobjective optimization problem $\bar{\mathcal{P}}_K$ induced by $K \in \mathcal{C}_1$, population size $N \in \mathbb{N}$, maximal amount of evaluations $M \in \mathbb{N}$, proper ascending cone mapping $\mathrm{c} : \mathcal{C}_1 \times D \to \mathcal{C}_2$, step size $\sigma \in \mathbb{R}_+$, perturbation factor $\delta \in D$ and threshold $\tau \in [0, 1]$. Appropriate selection, mutation and crossover operators.

1 **begin**

2 - Generate a parent population P_t of N random solutions $x_i \in \mathbb{R}^k$ and evaluate them.

3 - Set generation counter $t = 1$ and evaluation counter $e = N$.

4 **while** $e < M$ **do**

5 - Generate offspring population O_t of $\min\{N, M - e\}$ solutions using the selection, mutation and crossover operators.

6 - Evaluate O_t and set $e = e + \min\{N, M - e\}$.

7 - Set $R_t = P_t \cup O_t$.

8 - Perform algorithm 1 with input R_t, c, K, δ, τ and output P of the new parent population.

9 - Set $P_t = P$ and $t = t + 1$.

10 **end**

11 - Perform a non-dominated sorting to P_t w.r.t. the original cone K to identify the different fronts $\mathcal{F}_i, i = 1, 2, \ldots, l$, $l \in \mathbb{N}$.

12 - Perform algorithm 3 with input $\mathcal{F}_1, \mathrm{c}, K$ and σ.

13 **end**

Output: Approximation of a region of the efficient set $\mathcal{F}_1$ that covers $\mathcal{S}_{c(K,\delta)}$, is distributed w.r.t. τ and has assigned cone robustness degrees for all solutions $x \in \mathcal{F}_1$.

Here the selection, mutation and crossover operators which are used in line 5 to create a new offspring population are to be chosen out of the usual techniques used in `NSGA-II` which are applicable to select, modify and create solutions of a multiobjective optimization problem $\bar{\mathcal{P}}_K$. For a more detailed explanation of their functionality we recommend [7]. Note, that the non-dominated sorting algorithm used in line 8, 11 and 12 as well as the selection mechanism used in line 5 of `CREA`, takes care of violated constraints as well, whenever comparing two solutions. I.e. there exists a so called *constraint violation comparator*, which checks for feasibility of both solutions before

checking weather one solution $\hat{x} \in \mathbb{R}^n$ dominates another $\bar{x} \in \mathbb{R}^n$, i.e. if $\hat{x}, \bar{x} \in \mathcal{X} := \{x \in \mathbb{R}^k \mid g_i(x) \geq 0, \forall i \in \{1, 2, \ldots, l\}\}$ or not. If one solution is feasible and the other one not, the feasible one is preferred. If both are infeasible the constraint violation comparator prefers the solution with smaller *overall constraint violation*, which is defined to be $\left|\sum_{i=1}^{l} g_i(x)\right|$, for a solution $x \in \mathbb{R}^k \setminus \mathcal{X}$. Though if the for most problems stochastically very unlikely but possible case that $\left|\sum_{i=1}^{l} g_i(\bar{x})\right| = \left|\sum_{i=1}^{l} g_i(\hat{x})\right|$ holds true, the solutions $\bar{x}$ and $\hat{x}$ are incomparable w.r.t. the constraint violation comparator. Hence it is possible that there exists an index i s.t. the front $\mathcal{F}_i$ contains two infeasible solutions $\bar{x}, \hat{x} \in \mathbb{R}^n \setminus \mathcal{X}$ or more. Thus it is necessary to restrict ourselves onto problems $\bar{\mathcal{P}}_K$ having objective functions with domain $\mathbb{R}^n$ due to the fact, that the crowding distance assignment used in line 8 of algorithm 4 needs to calculate the objective function values of each solution handed in (see e.g. [2]).

2.1.2 Simulation Results

For our simulations we have been working with Metaheuristic Algorithms in Java or short `jMetal` [9], an object-oriented Java-based framework for multiobjective optimization with metaheuristics. This complex framework provides a whole bunch of completely implemented evolutionary and population based algorithms with their common and helpful components necessary for multiobjective evolutionary algorithms, such as evaluation and selection mechanisms as well as ranking, crossover and mutation methods. As part of this thesis we have extended version 4.5 of the `jMetal` framework by the Cone Robustness Based Evolutionary Algorithm and tested it on various test problems, including the ZDT problem family [46]. This family is composed of six carefully chosen multiobjective test problems containing features known to cause difficulties in the evolutionary optimzation process, such as multimodality and deception.

In order to test the `CREA` we have injected each test problem with perturbations within the cone $K \in \mathcal{K}^k$, defining the ordering relation $\preceq_K$ on $\mathbb{R}^k$ used in our optimization problem as in definition 1.5. Therefore we have been working with the proper ascending cone mapping $\bar{\mathrm{c}}$ of example 3, which maps $(\mathbb{R}^k_+, \delta)$ onto the polyhedral cone $K(A(\delta))$, where $A(\delta) \in \mathbb{R}^{k \times k}$ is the matrix primarily introduced in example 1. Notice that all methaheuristics presented in the `jMetal` framework only work with the strict partial order $\prec_{\mathbb{R}^k}$ induced by the Edgeworth-Pareto cone $\mathbb{R}^k_+$, hence they are implemented to find Edgeworth-Pareto optimal solutions only. We have extended

the `jMetal` framework by the `DegreeDominanceComparator` class, representing the strict partial order $\prec_{\hat{K}(\delta)}$ induced by the pointed polyhedral cone $\hat{K}(\delta) := K(A(\frac{\pi}{180}\delta))$ of example 1 with domain in arc degrees instead of radians. Doing so, we have presented the basic module enhancing all algorithms presented in `jMetal` in such a manner, that they become able to search for the efficient front w.r.t. $\hat{K}(\delta)$ of any multiobjective optimization problem $\mathcal{P}_{\hat{K}(\delta)}$ induced by $\hat{K}(\delta)$. The `DegreeDominanceComparator` class is explained more detailed in appendix E, which contains a descriptive list of all classes and interfaces implemented in order to include the new methaheuristics presented in this thesis, into the `jMetal` framework.
Since the transformation from radians into arc degrees does not interfere with any of the proofs given in example 1, 3 or 5, the mapping $\hat{\mathrm{c}}$ defined as

$$\hat{\mathrm{c}}: \ \mathcal{C}_1 \times \hat{D} \to \mathcal{C}_2, \ \ (\mathbb{R}^k_+, \delta) \mapsto \bar{\mathrm{c}}(\mathbb{R}^k_+, \frac{\pi}{180}\delta) = \hat{K}(\delta), \tag{12}$$

with $\hat{D} := \left[0, \frac{180}{\pi} \arctan\left(\frac{1}{\sqrt{k-1}}\right)\right)$, is a proper ascending cone mapping as well. We choose arc degrees over radians for our perturbation factor δ in order to keep it more vivid in the following simulation runs, especially when choosing small step sizes σ.

We have tested the `CREA` methaheuristic on each presented test problem and each presented adjustment of parameters for 30 simulation runs. For all problems solved we used a population of size $N = 300$ and a maximum of evaluations $M = 300000$. Hence the algorithm was running for 1000 generations. Furthermore we chose the simulated binary crossover (SBX) with distribution index 20, a crossover probability of 90%, the polynomial mutation operator with a probability of 50% and distribution index 20 and the binary tournament selection as selection operator to produce the offspring population in each generation(, see e.g. [7] for detailed descriptions).
The following examples are representative outcomes of these test runs, which mirror the properties the `CREA` possesses. The computer used for this evaluation has an Intel Core i5-3320M CPU with 2.6 GHz and 8 GiB installed memory as well as Windows 8 Pro N as operating system on a guest virtual machine whose host operating system is Ubuntu 14.04 LTS. The compact disc enclosed with this diploma thesis contains the implementation of the `CREA` integrated in the `jMetal` framework.
We have also implemented the macroinstruction `plotConeRobustnessDegree` enabling the decision maker to plot the outcome of a `CREA` run on a multidimensional optimization problem with two-dimensional objective space. It is

written in LaTeX under the usage of the `TikZ` and `pgfplots` packages [42], [15] and the file `Plot_CREA.tex` attached to the compact disk which is enclosed with this diploma thesis contains the instruction and a description within the comments (as well as an alpha version of the same macro able to plot three-dimensional outputs of the `CREA`).

For a presentation of such a produced vector graphic and a first simulation result, let us have a look back on example 2 depicted on figure 3. We have tested the `CREA` methaheuristic on example 2 with a step size $\sigma = 1°$, a perturbation factor $\delta = 20°$ and a threshold $\tau = 0$ as a first and $\tau = 1.0$ as a second adjustment of parameters. Remember that the choice of $\tau = 0$ results in a search for the whole Edgeworth-Pareto optimal front with an a posteriori assignment of the cone robustness degree. The choice of $\tau = 1.0$ results in a search attracted towards that part of the Edgeworth-Pareto optimal front which is cone robust efficient w.r.t. $(\hat{c}, 20°)$, though without any consideration of the crowding distance, as discussed after algorithm 1. Note that the domain $\hat{D}$ of the ascending cone mapping $\hat{c}$ used within our implementations equals the half-opened interval $[0, 45°)$ for $k = 2$. A representative outcome of the `CREA` on those parameter settings under usage of our `plotConeRobustnessDegree` macro is presented in figure 8.

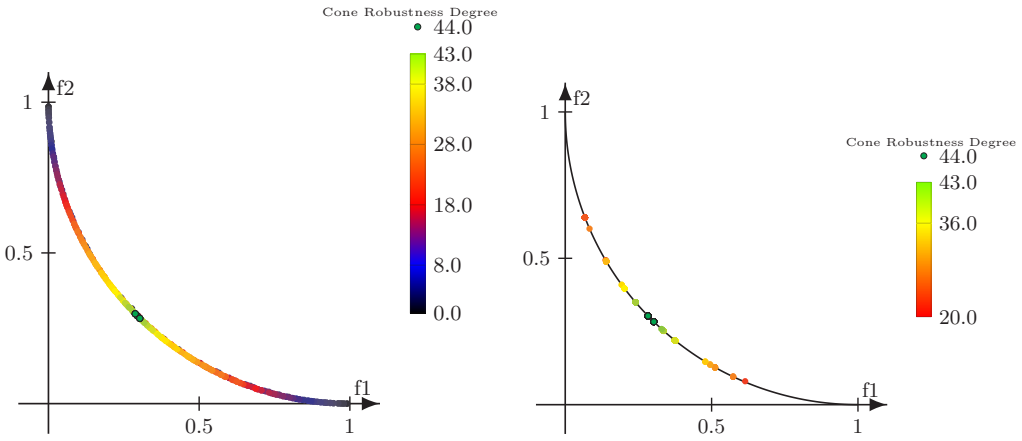


Figure 8: `CREA` on example 2, with $\tau = 0$ (left) and $\tau = 0.5$ (right)

As you can see in figure 8 the algorithm works as expected with the values 0 and 1.0 for our threshold τ on the test problem of example 2. For the second parameter setting depicted on the right, the part of the Edgeworth-Pareto optimal front was found, which has a cone robustness degree of at least 20°, though there is a very low diversity due to the extremal choice of $\delta = 1$. The

left of figure 8 depicts the whole Edgeworth-Pareto optimal front with cone robustness degree values between 0° and 44° assigned, where the assigned maximal cone robustness degree is 44° and not 45° due to the chosen step size of $\sigma = 1°$ and the convergence deliberations presented after algorithm 3 in equation (11). The following simulation results will depict those deliberations as the assigned cone robustness degree converges towards 45° if the chosen step size converges towards 0°. Furthermore, they will show that a lower choice of the threshold τ, leads to a higher diversity within the found set of efficient solutions.

Next we will present the results of the `CREA` simulation runs on the ZDT problem family introduced in [46]. On the figures to follow, which depict these results, the reference Edgeworth-Pareto optimal fronts provided on the `jMetal` homepage [10] are depicted in gray. We will start with the test problem ZDT1 presented in the next example.

Example 6
For $n \in \mathbb{N}$, the test problem ZDT1 is given through

$$\mathcal{P}_K : \quad \min{}_K \begin{pmatrix} x_1 \\ g(x)\left(1 - \sqrt{\frac{x_1}{g(x)}}\right) \end{pmatrix} \quad \text{s.t.} \quad x \in [0,1]^n,$$

with $g(x) := 1 + 9\frac{\left(\sum_{i=2}^{n} x_i\right)}{n-1}$.

Figure 9 depicts a representative output of the runs on example 6 with $n = 30$, a threshold $\tau = 0.5$ and a perturbation factor $\delta = 40°$, a step size $\sigma = 1°$ as a first and $\sigma = 0.1°$ as a second adjustment of parameters.

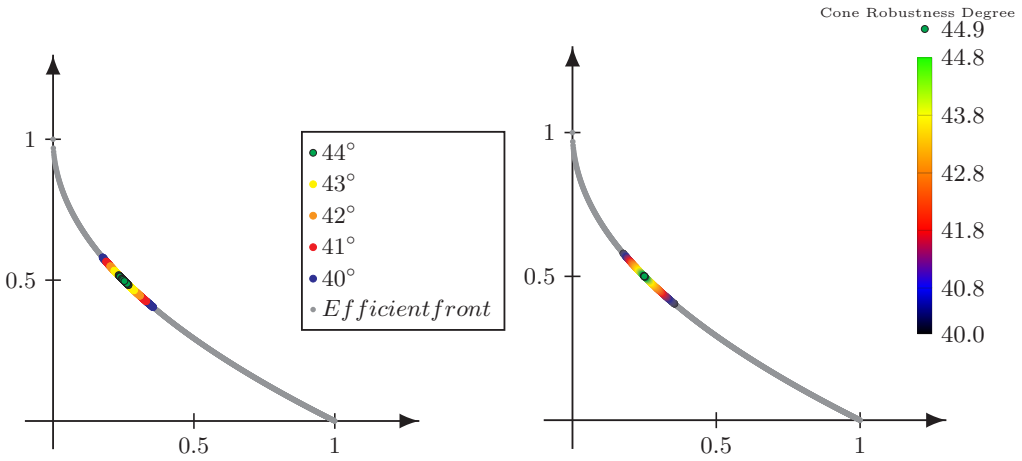


Figure 9: `CREA` on ZDT1, with $\sigma = 1°$ (left) and $\sigma = 0.1°$ (right)

Note, the left subfigure was not plotted via the `plotConeRobustnessDegree` instruction but each of the five cone robustness levels separately in order to depict the increasing accuracy of the cone robustness degree assignment under a decreasing step size σ clearly. In the representative run depicted on the left of figure 9, five cone robustness levels where found with minimal distance of $1°$ and on the run depicted on the right, 50 cone robustness levels with minimal distance of $0.1°$ where found. The next parameter setting on the example to follow reflects the two previous results but as well shows the importance of the included diversity mechanism of `CREA`.

Example 7
For $n \in \mathbb{N}$, the test problem ZDT2 has the following form.

$$\mathcal{P}_K: \quad \min{}_K \begin{pmatrix} x_1 \\ g(x)\left(1 - \left(\frac{x_1}{g(x)}\right)^2\right) \end{pmatrix} \quad \text{s.t.} \quad x \in [0,1]^n,$$

with $g(x) := 1 + 9\frac{\left(\sum_{i=2}^{n} x_i\right)}{n-1}$.

Figure 10 depicts a representative run on example 7 with $n = 30$, a perturbation factor $\delta = 28.5°$, a threshold $\tau = 0$ and step size $\sigma = 1°$ on a first and $\tau = 0.5$ and $\sigma = 0.1°$ on a second adjustment of parameters.

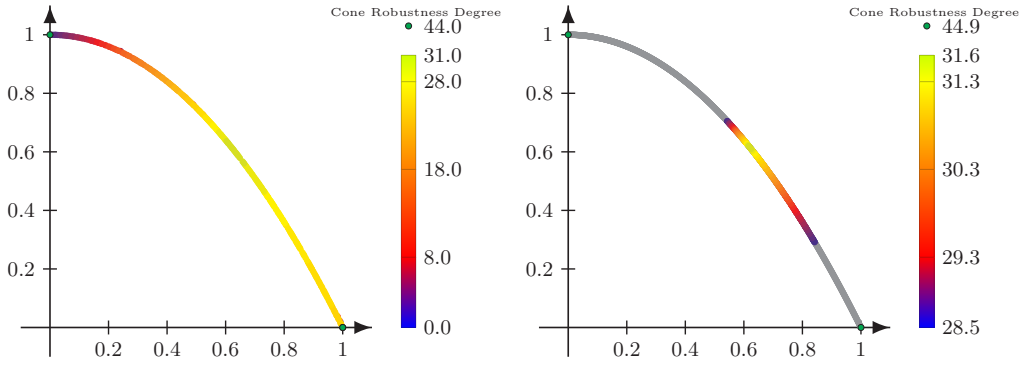


Figure 10: `CREA` on ZDT2, with $\tau = 0$ and $\sigma = 1°$ (left) and $\tau = 0.5$ and $\sigma = 0.1°$ (right)

Because of the concave form of the efficient front of ZDT2 the solutions with the maximal assigned cone robustness degree are the ones with objective values $(1, 0)$ and $(0, 1)$, as you can see on the left subfigure where we were searching for the whole Edgeworth-Pareto optimal front. In the second parameter adjustment we are searching for the solutions with cone robustness degree of at least $28.5°$. Note that this part of the efficient front is separated

from the two solutions with maximal cone robustness degree. Though, CREA is able to find these separated solutions, thanks to the diversity mechanism provided through the crowding distance assignment in algorithm 1 and an appropriate threshold setting of $\tau = 0.5$ ensuring a search for a well spread population directed towards the cone robust efficient set $\mathcal{S}_{\bar{c}(\mathbb{R}^k_+, 28.5°)}$, as depicted on the right of figure 10. The next adjustment of parameters on the last test problem of the ZDT family will underline this result.

Example 8
The test problem ZDT3 has the following form.

$$\mathcal{P}_K: \quad \min_K \begin{pmatrix} x_1 \\ g(x)\left(1 - \sqrt{\frac{x_1}{g(x)}} - \frac{x_1}{g(x)}\sin(10\pi x_1)\right) \end{pmatrix} \quad \text{s.t.} \quad x \in [0,1]^n,$$

with $g(x) := 1 + 9\frac{\left(\sum_{i=2}^n x_i\right)}{n-1}$.

Figure 11 depicts a representative output for the runs of CREA on example 8 with $n = 30$, a step size $\sigma = 0.5°$ a threshold $\tau = 0$ as a first adjustment of parameters and $\tau = 0$ such as perturbation factor $\delta = 5°$ as a second and $\delta = 20°$ as a third parameter setting.

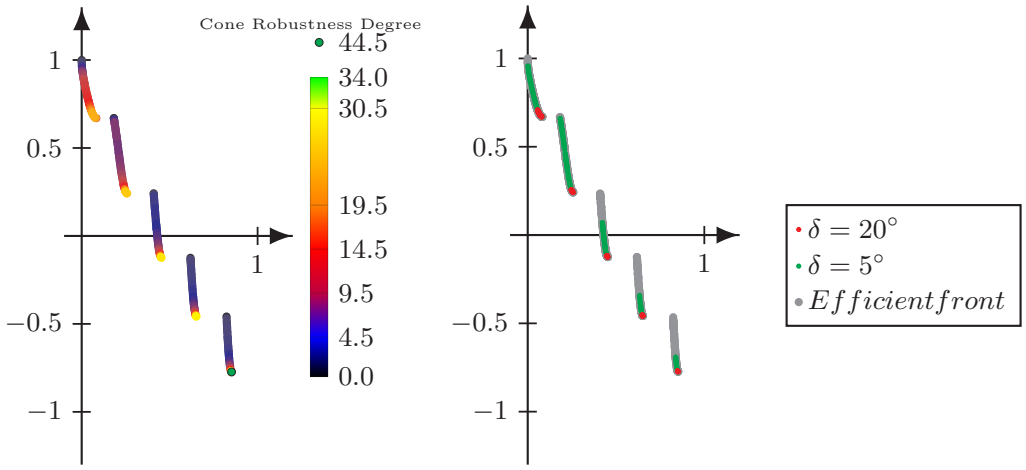


Figure 11: CREA on ZDT3 with $\tau = 0$(left), $\delta = 5°$ and $\delta = 20°$(right)

On the left of figure 11 one can see the output of a search for the whole efficient front $\mathcal{S}_K$ of ZDT3 with assigned cone robustness degrees, i.e. the output of the first adjustment of parameters. The right part of figure 11 depicts the obtained parts of the efficient front found for the second and third adjustment of parameters, which visualizes how the found front reduces towards

the cone robust efficient part our search is directed to, as the perturbation factor δ increases.

2.2 Conclusions

In this section we have developed a new definition of robustness for multiobjective optimization problems suffering under perturbations within the ordering inducing cone. In particular, we provide the definitions of an ascending cone mapping, of cone robust efficiency as well as the cone robustness degree measuring the maximal capability a solution stays undominated under perturbations of the ordering relation inducing cone. We have presented an evolutionary algorithm able to search for such robust efficient solutions as well as assigning each found solution its degree of robustness. Hence we have provided the powerful tool to solve black-box problems in such an order that a decision maker is able to find a set of efficient solutions robust under some level of uncertainties concerning the future preferences. As well, an assigned robustness degree gives him the possibility to estimate the risks lying within even higher perturbations.

Our work leaves many avenues for further research. First a deeper analysis of the cone robustness efficiency is desirable in order to prove analytically which part of a regarded efficient front is cone robust efficient w.r.t. the given ascending cone mapping setting. Such an analysis strongly depends on the currently regarded ascending cone mapping in connection with the shape and in especially the slope of the actual efficient front regarded. This topic was not addressed in this thesis, as our main focus was lying on a presentation of a general introduction into the theory of cone robustness rather than specifying too deeply on one single ascending cone mapping. A second aspect of future studies could be an integration of the cone robust efficiency tools within other evolutionary algorithms. The first step in this direction is already given through the implementation of the strict partial order $\prec_{\hat{K}(\delta)}$ and the assignment process of the cone robustness degree presented in algorithm 3. Hence, simply exchanging the strict partial order $\prec_{\mathbb{R}^k_+}$ induced by the Edgeworth-Pareto cone $\mathbb{R}^k_+$ through $\prec_{\hat{K}(\delta)}$ within other methaheuristics, will lead to evolutionary algorithms able to find cone robust efficient fronts. The inclusion of algorithm 3 into other methaheuristics will allow an assignment of cone robustness degrees or could be used in a ranking scheme applicable for arbitrary selection mechanisms. Also a deeper study of the Cone Robustness Evolutionary Algorithm himself with other ascending cone mappings than the one regarded here is of high interest. We have already implemented

an alpha version of such an extension for the `CREA` by providing the tools for opening each edge $E_i(\delta)$ of the regarded cone $K(A(\delta))$ of example 1 up to an individual maximum $\delta_i \in D$ for all $i \in \{1, 2, \ldots, k\}$ during the cone robustness assignment process of algorithm 3. This is implemented in the `CREA2` class, in the main class of which the user can give a double vector as input parameter to the constructor of the extended version `CREA2` instead of a single double value as given to `CREA`. Furthermore it is possible to extend the domain $\hat{D}$ of the implemented ascending cone mapping $\hat{\mathrm{c}}$ (presented in (12)) through regarding negative values for δ as well, as the proof of example 1 shows.

3 Uncertain Optimization Problems

If not denoted differently, throughout this section let $K \in \mathcal{K}^k$ be a proper closed and convex cone in $\mathbb{R}^k$. Hence together with corollary 1.6 it follows, that the binary relation $\preceq_K$ on $\mathbb{R}^k$ induced by K is a pre-order.

We are going to present a special class of multiobjective optimization problems $\mathcal{P}_K(U)$ which is infected with uncertainties concerning the objective map. These uncertainties are represented through an uncertainty set U consisting out of possible scenarios $\xi \in U$. Uncertain multiobjective optimization problems are very likely to occur in real world applications. For example, consider to buy a new car and you prefer the car with the highest comfort and lowest energy cost for the cheapest price. The last two objectives underlie the uncertainties of unknown future oil or electricity prices and what tax benefits you might get for owning an eco-friendly car, as you have to decide between gas or electricity drive [17]. Another example is given through finding an optimal traffic light circuit for a student city, where the decision maker has to analyze the geographic location and amount of possible cyclists, pedestrians, public transport and motorists and maybe even includes rush hour times into his calculations in order to adapt the circuit based on these results. Here uncertainties are likely to occur in all of the given parameters as students often change their residence, their means of transportation and also vary in their daily routine times, which leads to unpredictable traffic conditions.
An uncertain multiobjective optimization problem $\mathcal{P}_K(U)$ can analogously be formulated as a set-valued optimization problem. Thus in section 3.1 we are going to introduce various ordering relations used in set optimization which represent the decision makers preferences and risk-taking propensity. These ordering relations will then lead us to robustness definitions for our

uncertain multiobjective optimization problem in section 3.2. In section 3.3 we will present an evolutionary algorithm able to run on a special class of set-valued optimization problems or uncertain multiobjective optimization problems respectively. We will shortly explain the basic modules necessary for the implementation of this algorithm in section 3.3.2 and will present some numerical results in low dimensions in section 3.3.3.

As already stated in the literature review of this thesis, concerning the definitions within this section we will primarily orientate ourselves on the dissertation of J. Ide [17] from 2014, especially the paper [19] of Ide et al. covering the relationships between multi-objective robustness concepts and set-valued optimization was of importance. For the field of multiobjective set-valued optimization problems and the ordering relations stated here, we mainly orientated ourselves on the paper [24] of J. Jahn from 2010 where he introduced various new binary relations on power sets, nevertheless we will refer to other sources, if existent.

First we will need the definition of an uncertain multiobjective optimization problem, where the uncertain input data contaminates the formulation of a multiobjective optimization problem.

Definition 3.1
An uncertain multiobjective optimization problem $\mathcal{P}_K(U)$ *induced by* K *is given as the family* $(\mathcal{P}_K(\xi), \xi \in U)$ *of multidimensional optimization problems induced by* K*, with*

$$\mathcal{P}_K(\xi): \quad \min{}_K f(x,\xi) \quad s.t. \quad x \in \mathcal{X},$$

where $U \subseteq \mathbb{R}^m$ *is the* uncertainty set *consisting out of* scenarios $\xi \in U$*,* $f : \mathcal{X} \times U \to \mathbb{R}^k$ *is the objective function,* $\mathcal{X} \subseteq \mathbb{R}^n$ *represents the feasible set as usual and* $\mathcal{P}_K(\xi)$ *is an* instance of $\mathcal{P}_K(U)$*.*

And given $\mathcal{P}_K(U)$ *we define the* set of objective values of x *as*

$$f_U(x) := \{f(x,\xi) : \xi \in U\} \subseteq \mathbb{R}^k.$$

In the following we will only regard uncertain multiobjective optimization problems $\mathcal{P}_K(U)$ with non-empty uncertainty set U in order to gain non-empty sets of objective values $f_U(x)$ in $\mathscr{P}(\mathbb{R}^k)$ for all $x \in \mathcal{X}$. An uncertain

optimization problem $\mathcal{P}_K(U)$ can as well be written down as a set-valued optimization problem $\mathcal{SP}_K^U$, which will be shown in the next subsection.

3.1 Set-Valued Optimization

Within this thesis we will regard set-valued optimization problems of the following form.

Definition 3.2
Let $\preccurlyeq$ be a pre-order on the power set $\mathscr{P}(\mathbb{R}^k)$ of $\mathbb{R}^k$, then the multiobjective set-valued optimization problem induced by $\preccurlyeq$ *is given by*

$$\mathcal{SP}_{\preccurlyeq}: \quad \min{}_{\preccurlyeq} F(x) \quad s.t. \quad x \in \mathcal{X},$$

where $F : \mathcal{X} \to \mathscr{P}(\mathbb{R}^k)$ is called a set-valued objective map *and is also denoted as $F : \mathcal{X} \rightrightarrows \mathbb{R}^k$, $\mathcal{X} \subseteq \mathbb{R}^n$ is the feasible set and $F(\mathcal{X}) = \bigcup_{x\in\mathcal{X}} F(x)$ is the objective space of $\mathcal{SP}_{\preccurlyeq}$.*

We will also need a special class of set-valued optimization problems, to which we will refer to as parametrized set-valued optimization problem.

Definition 3.3
Let $\preccurlyeq$ be a pre-order on the power set $\mathscr{P}(\mathbb{R}^k)$ of $\mathbb{R}^k$, then the parametrized multiobjective set-valued optimization problem induced by $\preccurlyeq$ *is given by*

$$\mathcal{SP}_{\preccurlyeq}^S: \quad \min{}_{\preccurlyeq} F(x) \quad s.t. \quad x \in \mathcal{X},$$

where the set-valued objective map

$$F : \mathcal{X} \to \mathscr{P}(\mathbb{R}^k), \;\; x \mapsto \{p(x,\xi) \in \mathbb{R}^k \,|\, \xi \in S\}$$

is described by a parametrized objective function $p : \mathcal{X} \times S \to \mathbb{R}^k$, *with* parameter set $S \subseteq \mathbb{R}^m$ *containing all parameters $\xi \in S$ of $\mathcal{SP}_{\preccurlyeq}^S$, $\mathcal{X} \subseteq \mathbb{R}^n$ is the feasible set and $F(\mathcal{X}) = \bigcup_{x\in\mathcal{X}} F(x)$ is the objective space of $\mathcal{SP}_{\preccurlyeq}^S$.*

Since minimizing on a power set is not totally intuitive, we will have to give an efficiency definition for mutliobjective set-valued optimization problems as well.

Definition 3.4 (Efficiency)
Let $\preccurlyeq$ be a pre-order on $\mathscr{P}(\mathbb{R}^k)$ and let $\mathcal{SP}_{\preccurlyeq}$ be the multiobjective set-valued

optimization problem induced by $\preccurlyeq$*, then* $x^0 \in \mathcal{X}$ *is called an* efficient *solution w.r.t.* $\preccurlyeq$ *of* $\mathcal{SP}_{\preccurlyeq}$ *iff*

$$F(x) \preccurlyeq F(x^0) \text{ for some } x \in \mathcal{X} \implies F(x^0) \preccurlyeq F(x).$$

The efficient set of $\mathcal{SP}_{\preccurlyeq}$ *is denoted by* $\mathcal{S}_{\preccurlyeq} := \{x \in \mathcal{X} \mid x \text{ is efficient w.r.t. } \preccurlyeq\}$ *and the efficient front of* $\mathcal{SP}_{\preccurlyeq}$ *then is defined to be* $\mathcal{E}(\mathcal{SP}_{\preccurlyeq}) := \bigcup_{x \in \mathcal{S}_{\preccurlyeq}} F(x)$.

Notice that there exist a lot of different solution concepts in set-valued optimization, such as the concepts based on a vector-approach or the ones based on a lattice-structure. The above given concept based on a set-approach is probably the most common one and more suitable from a practical point of view than others (see e.g. [3], [19], [24], [26]). For an overview of the varieties of existing solution concepts and a deeper insight on the basics of set-valued optimization in general, we recommend [26].

Given an uncertain optimization problem $\mathcal{P}_K(U)$ and a pre-order $\preccurlyeq$ on $\mathscr{P}(\mathbb{R}^k)$, the parametrized set-valued optimization problem $\mathcal{SP}^S_{\preccurlyeq}$ with parameter set $S := U$, set-valued objective map $F(x) := f_U(x)$ for all $x \in \mathcal{X}$ and parametrized objective function $p(x, \xi) := f(x, \xi)$ for all $x \in \mathcal{X}$ and $\xi \in U$ represents the uncertain optimization problem $\mathcal{P}_K(U)$ and will be denoted as

$$\mathcal{SP}^U_{\preccurlyeq}: \quad \min f_U(x) \quad \text{s.t.} \quad x \in \mathcal{X}. \tag{13}$$

Notice that the here given pre-order $\preccurlyeq$ plays a key role, as it will define our robustness notion for our uncertain optimization problem $\mathcal{P}_K(U)$, as the next sections will show.

3.1.1 Existing Binary Relations

In the following, the sets A and B are elements of the power set $\mathscr{P}(\mathbb{R}^k)$.

In this section we are going to introduce seven existing binary relations on $\mathscr{P}(\mathbb{R}^k)$ known in the field of set-valued optimization. In order to keep it more lucid we will present the most common notation, for each relation, in literature in the header of each definition respectively.

Definition 3.5 (Lower set less order relation '$\preccurlyeq^l_K$')

$$A \preccurlyeq^l_K B \quad :\Longleftrightarrow \quad (\forall b \in B \ \exists a \in A : a \preceq_K b) \quad \Longleftrightarrow \quad A + K \supseteq B.$$

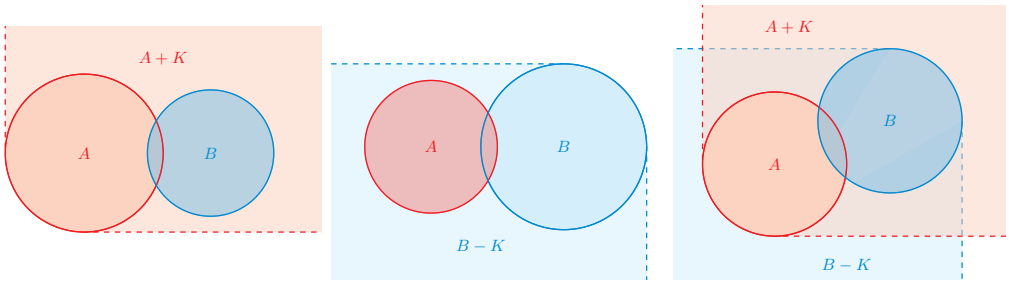
Definition 3.6 (Upper set less order relation '$\preccurlyeq^u_K$')

$$A \preccurlyeq^u_K B \quad :\Longleftrightarrow \quad (\forall a \in A \ \exists b \in B : a \preceq_K b) \quad \Longleftrightarrow \quad A \subseteq B - K.$$

The upper and lower set less order relations have been presented by Kuroiwa in [30] and [31] and have been subject of study for various mathematicians since then ([18], [24], [26]). Figure 12 depicts the first two relations. Note that in the subfigure (a) $A \preccurlyeq_K^l B$ holds true but not $A \preccurlyeq_K^u B$ as well as in subfigure (b) $A \preccurlyeq_K^u B$ holds true but not $A \preccurlyeq_K^l B$. Figure 12 (c) shows the following binary relation on sets, which can be described through the previous two and has been introduced independently by Young [45] and Nishnianidze [34].

Definition 3.7 (Set less order relation '$\preccurlyeq_K^s$')

$$A \preccurlyeq_K^s B \quad :\Longleftrightarrow \quad A \preccurlyeq_K^u B \quad \textit{and} \quad A \preccurlyeq_K^l B.$$



(a) $A \preccurlyeq_K^l B$ (b) $A \preccurlyeq_K^u B$ (c) $A \preccurlyeq_K^s B$

Figure 12: Lower, upper set less order relation and set less order relation

A much stronger binary relation is given through the 'certainly less' relation, which was introduced by Chiriaev and Walster in [5] and is depicted in figure 13 (a).

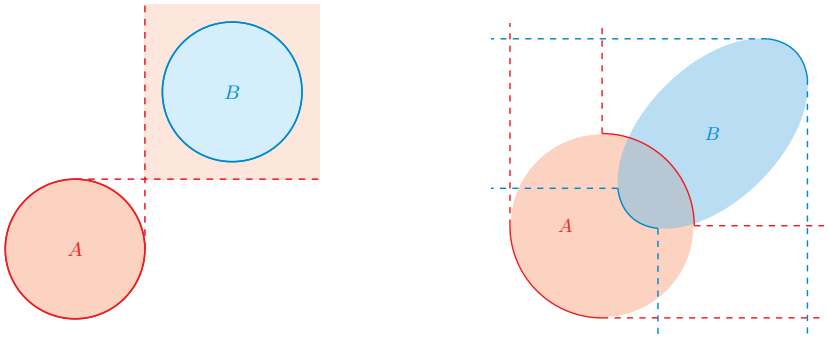
Definition 3.8 (Certainly less order relation '$\preccurlyeq_K^c$')

$$A \preccurlyeq_K^c B \quad :\Longleftrightarrow \quad (A = B) \quad or \quad (\forall a \in A \quad \forall b \in B : \quad a \preceq_K b).$$

The following three order relations were primarily introduced by Jahn and Ha in [24] in 2010. Since they are working with the minimal and maximal elements w.r.t. K of the sets to compare, from now on it holds true that $A, B \in \mathscr{M} := \{A \in \mathscr{P}(\mathbb{R}^k) \mid \min_K A \neq \emptyset, \max_K A \neq \emptyset\}$, where $\min_K A$ or $\max_K A$ are the efficient front w.r.t. K of the multiobjective optimization problem $\mathcal{P}_K$ with feasible set $\mathcal{X} := A$ and objective map $f := id^k$ or $f := -id^k$ respectively.

Definition 3.9 (Minmax less order relation '$\preccurlyeq_K^m$')

$$A \preccurlyeq_K^m B \quad :\Longleftrightarrow \quad \min_K A \preccurlyeq_K^s \min_K B \quad \textit{and} \quad \max_K A \preccurlyeq_K^s \max_K B.$$



(a) $A \preccurlyeq_K^c B$ (b) $A \preccurlyeq_K^m B$

Figure 13: Certainly and minmax less order relation

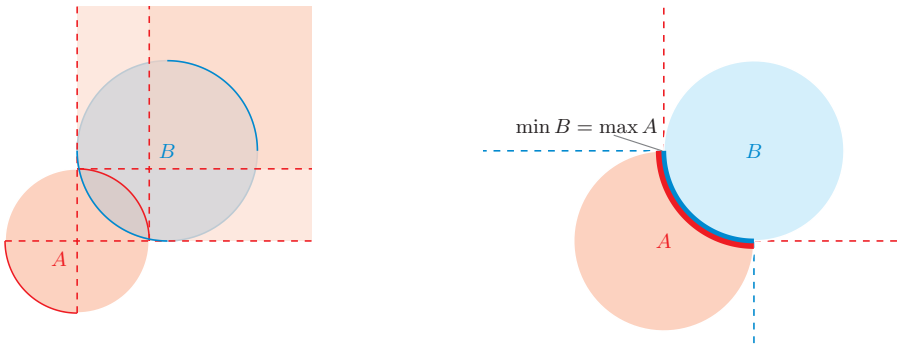
Definition 3.10 (Minmax certainly less order relation '$\preccurlyeq_K^{mc}$')

$$A \preccurlyeq_K^{mc} B \quad :\Longleftrightarrow \quad (A = B) \quad or \\ (A \neq B, \quad \min_K A \preccurlyeq_K^c \min_K B \quad and \quad \max_K A \preccurlyeq_K^c \max_K B).$$

Definition 3.11 (Minmax certainly nondominated order relation '$\preccurlyeq_K^{mn}$')

$$A \preccurlyeq_K^{mn} B \quad :\Longleftrightarrow \quad (A = B) \quad or \quad (A \neq B, \quad \max_K A \preccurlyeq_K^s \min_K B).$$

Figure 14 depicts the last two order relations regarded in this thesis. Note that there exists no equality between the certainly less order relation $\preccurlyeq_K^c$ and the minmax certainly nondominated order relation $\preccurlyeq_K^{mn}$ as you can see on the right subfigure.



(a) $A \preccurlyeq_K^{mc} B$ (b) $A \preccurlyeq_K^{mn} B \;\&\; \neg(A \preccurlyeq_K^c B)$

Figure 14: Minmax certainly less and nondominated order relation

The following result is of importance in order to use the concerning binary relations on set-valued optimization problems as the next section will prove.

Theorem 3.12
Let $K \in \mathcal{K}^k$ be a proper, convex and closed cone in $\mathbb{R}^k$. Then it follows that the binary relation $\preccurlyeq_K^*$ is a pre-order

(i) on $\mathscr{P}(\mathbb{R}^k)$, for all $* \in \{l, u, s, c\}$,

(ii) on $\mathscr{M}$, for all $* \in \{m, mc, mn\}$.

Proof: Assertion (i) is proven in proposition 3.1 and 3.2 of [24] and (ii) in poposition 3.4 and 3.7 of [24] respectively. ■

3.2 Definition of Robustness

From here on and if not denoted differently, whenever we are regarding an uncertain multiobjective optimization problem $\mathcal{P}_K(U)$ or a set-valued optimization problem $\mathcal{SP}_{\preccurlyeq_K^*}$, we are going to distinguish between the underlying pre-order $\preccurlyeq_K^*$ with $* \in \{l, u, s, c, m, mc, mn\}$ in the following manner. If $* \in \{m, mc, mn\}$ holds true, only uncertain optimization problems $\mathcal{P}_K(U)$ are regarded that have the following properties: Their corresponding uncertainty sets U and objective functions f fulfill $f_U(x) \in \mathscr{M}$ for all $x \in \mathcal{X}$ and respectively, only set-valued optimzization problems are regarded with $F(x) \in \mathscr{M}$ for all $x \in \mathcal{X}$.

In [19] of Ide et al. the following three definitions of robustness for our uncertain optimization problem $\mathcal{P}_K(U)$ were presented.

Definition 3.13 (Definition 6, 7 and 8 of [19])
Given an uncertain multiobjective optimization problem $\mathcal{P}_K(U)$ a solution $x^0 \in \mathcal{X}$ is $\preccurlyeq_K^*$-robust for $\mathcal{P}_K(U)$ iff there exists no other solution $x \in \mathcal{X} \setminus \{x^0\}$ s.t. $f_U(x) \preccurlyeq_K^* f_U(x^0)$, where $* \in \{l, u, s\}$.

Note that in [19] the authors were working with an arbitrary linear topological space Y partially ordered by a pointed closed convex and not blunt cone K, instead of the linear space $\mathbb{R}^k$ we confined ourselves to and they extended the above given robustness definition by allowing to replace K with its blunt version $K \setminus \{0\}$ and its interior $int\{K\}$ as well.

Within the conclusions of [19] (p. 18) the authors state the following assertion we will refer to as assertion (A).

Given one of the three pre-orders $\preccurlyeq_K^*$ on $\mathscr{P}(\mathbb{R}^k)$ with $* \in \{u, l, s\}$, solving the set-valued optimization problem $\mathcal{SP}^U_{\preccurlyeq_K^*}$ is equivalent to finding $\preccurlyeq_K^*$-robust solutions to the uncertain multiobjective optimization problem $\mathcal{P}_K(U)$.

This assertion in fact is wrong if efficiency for a set-valued optimzation problem is defined as in definition 3.4 or definition 4 of [19] respectively, as the next example will prove.

Example 9
Let us regard the set-valued optimzation problem $\mathcal{P}^U_{\preccurlyeq_K^u}$ induced by the upper set less order relation $\preccurlyeq_K^u$ representing an uncertain optimization problem $\mathcal{P}_K(U)$ as in equation (13), with feasible set $\mathcal{X}$ of cardinality 2, i.e. let $\mathcal{X} := \{x_1, x_2\}$, with $x_1, x_2 \in \mathbb{R}^n$. And let $f_U(x_1) - K = f_U(x_2) - K$ and $f_U(x_1) \neq f_U(x_2)$ hold true. Then obviously $f_U(x_1) \preccurlyeq_K^u f_U(x_2)$ as well as $f_U(x_1) \preccurlyeq_K^u f_U(x_2)$ holds true, as depicted in figure 15 for $K := \mathbb{R}^k_+$. Hence x_1 as well as x_2 are both efficient to $\mathcal{P}^U_{\preccurlyeq_K^u}$, though no one of them is $\preccurlyeq_K^u$-robust for $\mathcal{P}_K(U)$.

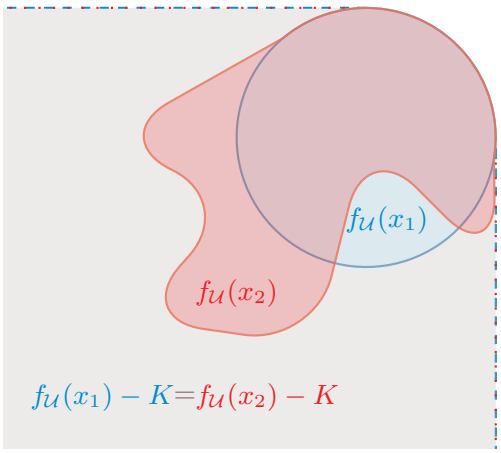


Figure 15: Illustration of example 9

Note that, as done by J. Ide in [17], by changing the definition of efficiency for a set-valued optimization problem from definition 3.4 to the following alternative, the assertion (A) in [19] obviously becomes true.

Definition 3.14 (of [17] page 10)
Let $\precsim$ be a binary relation on $\mathscr{P}(\mathbb{R}^k)$ and let $\mathcal{SP}_{\precsim}$ be the multiobjective set-

valued optimization problem induced by $\precsim$, then $x^0 \in \mathcal{X}$ is called an efficient *solution to $\mathcal{SP}_{\precsim}$ iff there exists no $x \in \mathcal{X} \setminus \{x^0\}$ such that $F(x) \precsim F(x^0)$.*

Here J. Ide as well had to change the definition of a set-valued optimization problem to one induced by an arbitrary binary relation instead of a pre-order, since replacing the cone K, as annotated after definition 3.13, will change the properties of the pre-orders used in this definition in such a way, that it is no pre-order anymore. E.g. regarding the upper set less order relation $\preccurlyeq_K^u$, changing K to its interior $int\{K\}$ will lead to an irreflexive binary relation.

In order to stay consistent with the literature of set-valued optimization problems we will stick to the prevalent efficiency definition 3.4 for set-valued optimization problems. Though, the search for $\preccurlyeq_K^*$-robust solutions as defined in definition 3.13, with $* \in \{u, l, s\}$, would now restrict our search for robust solutions of the uncertain optimization problem $\mathcal{P}_K(U)$, in such a manner, that we wont be able to find solutions as presented in example 9. This would prevent the decision maker from having the opportunity to choose out of a broader set of robust solutions. Hence in order to avoid this restriction and stay consistent with the definition of robustness presented in [19] as well, we have extended definition 3.13 to a more general setting.

Definition 3.15 ($\precsim$-robust efficiency)
Given an uncertain multiobjective optimization problem $\mathcal{P}_K(U)$ and an arbitrary ordering relation $\precsim$ on $\mathscr{P}(\mathbb{R}^k)$, a solution $x^0 \in \mathcal{X}$ is called $\precsim$-robust efficient *for $\mathcal{P}_K(U)$ iff there exists no other solution $x \in \mathcal{X} \setminus \{x^0\}$ s.t. $f_U(x) \precsim f_U(x^0)$.*

In order to find the complete efficient set $\mathcal{S}_{\preccurlyeq}$ of a set-valued optimization problem $\mathcal{SP}_{\preccurlyeq}$ we will need the following definitions of strict partial orders induced by the pre-orders presented in the last subsection.

Definition 3.16
Let $* \in \{l, u, s, c, m, mc, mn\}$ and $A, B \in \mathscr{P}(\mathbb{R}^k)$, then the strict partial order $\prec_K^*$ induced by $\preccurlyeq_K^*$ on $\mathscr{P}(\mathbb{R}^k)$ is defined as

$$A \prec_K^* B \iff A \preccurlyeq_K^* B \text{ and } \neg(B \preccurlyeq_K^* A).$$

Lemma 1.3 directly proves with theorem 3.12 that $\prec_K^*$ really is a strict partial order on $\mathscr{P}(\mathbb{R}^k)$ for all $* \in \{l, u, s, c, m, mc, mn\}$. The following theorem presents an overview of the connection between uncertain optimization problems and set-valued optimization.

Theorem 3.17
Let $\mathcal{P}_K(U)$ be an uncertain multiobjective optimization problem, let the set-valued optimization problem $\mathcal{SP}^U_{\preccurlyeq_K^*}$ be defined as in equation (13) *for all*

$* \in \{l, u, s, c, m, mc, mn\}$ and let $x^0 \in \mathcal{X}$ be a solution of $\mathcal{P}_K(U)$ and $\mathcal{SP}_{\preccurlyeq^U_{K*}}$ respectively, then the following holds true for all $* \in \{l, u, s, c, m, mc, mn\}$.

$$
\begin{aligned}
& x^0 \text{ is efficient for } \mathcal{SP}^U_{\preccurlyeq^*_K} && (14)\\
\iff\ & x^0 \text{ is } \prec^*_K \text{-robust efficient for } \mathcal{P}_K(U) && (15)\\
\impliedby\ & x^0 \text{ is } \preccurlyeq^*_K \text{-robust efficient for } \mathcal{P}_K(U). && (16)
\end{aligned}
$$

Proof: Let $* \in \{l, u, s, c, m, mc, mn\}$.

(14) $\iff$ (15):

$$
\begin{aligned}
& x^0 \text{ is efficient for } \mathcal{SP}^U_{\preccurlyeq^*_K}\\
\iff\ & \forall x \in \mathcal{X} \text{ with } f_U(x) \preccurlyeq^*_K f_U(x^0) \text{ follows } f_U(x^0) \preccurlyeq^*_K f_U(x)\\
\iff\ & \nexists x \in \mathcal{X} : f_U(x) \preccurlyeq^*_K f_U(x^0) \text{ and } \neg(f_U(x^0) \preccurlyeq^*_K f_U(x))\\
\iff\ & \nexists x \in \mathcal{X} : f_U(x) \prec^*_K f_U(x^0)\\
\iff\ & x^0 \text{ is } \prec^*_K \text{-robust efficient.}
\end{aligned}
$$

(15) $\impliedby$ (16):

$$
\begin{aligned}
& x^0 \text{ is } \preccurlyeq^*_K \text{-robust efficient } x^0\\
\iff\ & \nexists x \in \mathcal{X} : f_U(x) \preccurlyeq^*_K f_U(x^0)\\
\implies\ & \nexists x \in \mathcal{X} : f_U(x) \preccurlyeq^*_K f_U(x^0) \text{ and } \neg(f_U(x^0) \preccurlyeq^*_K f_U(x))\\
\iff\ & x^0 \text{ is } \prec^*_K \text{-robust efficient } x^0.
\end{aligned}
$$

■

At last we have to give a definition of domination for set-valued optimization problems in order to describe the algorithms of the next section properly.

Definition 3.18 (Domination)
Let $\mathcal{SP}_{\preccurlyeq^*_K}$ be a multiobjective set-valued optimization problem and $\prec^*_K$ defined as in definition 3.16, with $* \in \{l, u, s, c, m, mc, mn\}$. If $f(\bar{x}) \prec^*_K f(x)$ holds true for two feasible solutions $\bar{x}, x \in \mathcal{X}, \bar{x} \neq x$, then the solution x is called dominated by $\bar{x}$ w.r.t. $\prec^*_K$ *and synonymously it is said, that $\bar{x}$* dominates x w.r.t. $\prec^*_K$. *If for two arbitrary feasible solutions $x, \bar{x} \in \mathcal{X}, \bar{x} \neq x$ neither $f(x) \prec_K f(\bar{x})$ nor $f(\bar{x}) \prec_K f(x)$ is fulfilled, they are called* incomparable w.r.t. $\prec^*_K$ *to one another.*

3.3 Computational Methods

As done in subsection 2.1, here again we confine ourselves to a special class of optimization problems in order to guarantee the functionality of all tools used in the algorithm to follow.

For $* \in \{l, u, s, c, m, mc, mn\}$ we regard the following class of parametrized set-valued multiobjective optimization problems.

$$\mathcal{SP}^{S}_{\preccurlyeq^*_K}: \quad \min{}_{\preccurlyeq^*_K} F(x) \quad \text{s.t.} \quad x \in \mathcal{X},$$

with set-valued objective map $F : \mathbb{R}^n \to \mathscr{P}(\mathbb{R}^k),\ x \mapsto \{p(x, \xi) \,|\, \xi \in S\}$, parametrized objective function $p : \mathbb{R}^n \times S \to \mathbb{R}^k$, parameter set $S \subseteq \mathbb{R}^m$ and feasible set $\mathcal{X} := \left\{x \in \mathbb{R}^n \,|\, g_i(x) \leq 0, \forall i \in \{1, 2, \ldots, l\}\right\}$, with inequality constraint functions $g_i : \mathbb{R}^n \to \mathbb{R}$ for all $i \in \{1, 2, \ldots, l\}$.

Here the set-valued objective map F has the domain $\mathbb{R}^n$ in order to guarantee the applicability of our crowding distance comparator and the feasible set $\mathcal{X}$ is given through inequality constraints in order to allow the usage of the constraint violation comparator as described in section 2.1.1.

To our best knowledge, there does not exist any attempt so far on using population based heuristics or evolutionary algorithms in general for solving any kind of multiobjective set-valued optimization problems or uncertain multiobjective optimization problems. As part of this diploma thesis we have extended the open source Java framework `jMetal` [9] by the set-valued non-dominated sorting based genetic algorithm II, which is able to run on both types of optimization problems due to the equivalence presented in theorem 3.17. This presents an introduction for evolutionary algorithms into the field of multiobjective set-valued optimization and uncertain multiobjective optimization respectively.

3.3.1 Set-Valued Non-Dominated Sorting Based Genetic Algorithm II (`SV-NSGA-II`)

From here on we will restrict our formulations on parametrized set-valued optimization problems in the interest of clarity. Note that all of the results to follow are as well applicable onto uncertain multiobjective optimization problems, due to theorem 3.17.

The SV-NSGA-II has the same basic structure as the NSGA-II [2], but is able to run on parametrized multiobjective set-valued optimization problems. This opens a completely new field of application for evolutionary algorithms on set-valued optimization problems which comes with a bunch of questions as well as opportunities. Regarding the SV-NSGA-II we had to face the question on how to design the basic modules within jMetal in order to generate a methaheuristic able to run on parametrized set-valued multiobjective optimization problems. Also we had to find a way how to construct the selection mechanism and the crowding distance assignment and regarding set-valued optimization problems in general one has to find an applicable and appropriate way to model the problems and sample the images of the set-valued objective function. In the following we are going to elucidate, how we answered those questions and why we chose the way we did.

As handled in section 2.1.1, here again we are assuming a population size of $N \in \mathbb{N}$ for our parent population P_t and offspring population O_t in the following, i.e. $|P_t| = N = |O_t|$. Let $R_t := P_t \cup O_t$ be their union, at any *generation* $t \in \mathbb{N}$, such as $e \in \mathbb{N}$ a counter for the current amount of evaluations of the set-valued objective function F of $\bar{\mathcal{SP}}^S_{\preccurlyeq^*_K}$ and let $M \in \mathbb{N}$ be the maximal amount of evaluations, defining the limit of generations SV-NSGA-II is running for.
Given a parametrized multiobjective set-valued optimization problem $\bar{\mathcal{SP}}^S_{\preccurlyeq^*_K}$ with $* \in \{l, u, s, c, m, mc, mn\}$, the SV-NSGA-II is able to search for all efficient solutions of $\bar{\mathcal{SP}}^S_{\preccurlyeq^*_K}$ in one run. Algorithm 5 presents its framework.

Algorithm 5 : `SV-NSGA-II`

Input: Proper closed convex and pointed cone $K \subset \mathbb{R}^k$, pre-order $\preccurlyeq_K^*$ with $* \in \{l, u, s, c, m, mc, mn\}$, parametrized multiobjective set-valued optimization problem $\mathcal{SP}^S_{\preccurlyeq_K^*}$, amount of sample points $s \in \mathbb{N}$ population size $N \in \mathbb{N}$, maximal amount of evaluations $M \in \mathbb{N}$, selection, mutation and crossover operators suitable for $\mathcal{SP}^S_{\preccurlyeq_K^*}$.

begin

- Generate a parent population P_t of N random solutions and evaluate them.
- Set generation counter $t = 1$ and evaluation counter $e = N$.
  while $e < M$ **do**
    - Generate offspring population O_t of $\min\{N, M - e\}$ solutions using the appropriate selection, mutation and crossover operators.
    - Evaluate O_t and set $e = e + \min\{N, M - e\}$.
    - Set $R_t = P_t \cup O_t$.
    - Perform a non-dominated sorting to R_t w.r.t. $\prec_K^*$ to identify the different fronts $\mathcal{F}_i, i = 1, 2, \ldots, l,\ l \in \mathbb{N}$.
    - Set $j = 1$ and $P = \emptyset$.
    while $|P \cup \mathcal{F}_j| < N$ **do**
      - Assign crowding distance to all elements of $\mathcal{F}_j$ using an appropriate crowding distance assignment.
      - Replace P by $P \cup \mathcal{F}_j$.
      - Replace j by $j + 1$.
    end
    - Assign crowding distance to all elements of $\mathcal{F}_j$ using an appropriate crowding distance assignment.
    - Add $N - |P|$ solutions of $\mathcal{F}_j$ to P considering their assigned crowding distance values.
    - Set $P_t = P$ and $t = t + 1$.
  end
- Perform a non-dominated sorting to P_t w.r.t. $\prec_K^*$ to identify the different fronts $\mathcal{F}_i, i = 1, 2, \ldots, l,\ l \in \mathbb{N}$.

end

Output: Approximation of the efficient set $\mathcal{S}_{\preccurlyeq_K^*}$ of $\mathcal{SP}^S_{\preccurlyeq_K^*}$.

Note that we do not have to explicitly submit the strict partial order $\prec_K^*$ as an input parameter of algorithm 5 since $\prec_K^*$ is induced by the pre-order $\preccurlyeq_K^*$ as presented in definition 3.16. The amount of sample points $s \in \mathbb{N}$ submitted as input parameter defines the number of samples to take of the parameter set $S \subseteq \mathbb{R}^m$ for each solution $x \in \mathcal{X}$, i.e. it defines the discretization $S_D := \{\xi^1, \xi^2, \ldots, \xi^s\}$ of S. Thus, this parameter is of importance as it defines how adequately the image $F(x)$ is represented and hence how good the comparison of two solutions under the usage of the chosen cone K and pre-order $\preccurlyeq_K^*$ is approximated. Also it defines the amount of comparisons necessary in order to test wether one solution $x \in \mathcal{X}$ dominates another solution $\bar{x} \in \mathcal{X}$ in dependence of the strict partial order $\prec_K^*$ we are working with as described later in section 3.3.3. Hence the choices of parameter $s \in \mathbb{N}$ as well as the choice of the pre-order $\preccurlyeq_K^*$ has a high influence on the running time of the algorithm.

Since the mutation and crossover mechanisms suitable for deterministic multiobjective optimization problems $\mathcal{P}_K$ are working in the decision space $\mathcal{X}$ and not in the objective space Y, they are usable for parametrized multiobjective set-valued optimization problems $\mathcal{SP}^S_{\preccurlyeq_K^*}$ in general as well. The selection mechanism on the other hand has to be adapted in order to run on the set-valued objective space $F(\mathcal{X})$. We chose to adjust the *binary tournament selection* presented by K. Deb in [7] within our implementations. Here two solutions $x, \bar{x}$ get picked out of the parent population P_t in generation t of the `SV-NSGA-II` at random. If one of the solutions dominates the other one, with respect to the chosen strict partial order $\prec_K^*$, the dominating solution will be returned. If they are incomparable to each other w.r.t. $\prec_K^*$ the solution with lower crowding distance will be selected and if the crowding distance of both solutions are equal or not assigned yet, as in the first iteration of the algorithm, one of the two solutions $x, \bar{x}$ is selected at random. In order to assign a crowding distance to each element of a set $\mathcal{F} \subseteq \mathcal{X}$ of solutions of a parametrized set-valued optimization problem $\mathcal{SP}^S_{\preccurlyeq_K^*}$ the *representative crowding distance assignment* picks for each solution $x \in \mathcal{F}$ a representative point $p(x, \hat{\xi}) \in F(x)$ out of its image set $F(x)$ and performs a classical crowding distance assignment on the obtained set of representative points as presented in detail in algorithm 6.

Algorithm 6 : Representative crowding distance assignment in `SV-NSGA-II`

Input: Parametrized objective function $p : \mathbb{R}^n \times S \to \mathbb{R}^k$, representative parameter $\hat{\xi} \in S$, $* \in \{l, u, s, c, m, mc, mn\}$ set $\mathcal{F} := \{x^1, x^2, \ldots, x^l\}$ of $l \in \mathbb{N}$ solutions $x^i \in \mathbb{R}^n$, $i \in \{1, 2 \ldots, l\}$.

begin
  if $l \leq 2$ **do**
    - Assign to each solution in $\mathcal{F}$ a crowding distance of ∞.
  else
    - Set counter $i = 1$, set representative set $\mathcal{I} = \emptyset$.
    while $i \leq l$
      - Add representative $(x^i, \hat{\xi})$ to $\mathcal{I}$.
      - Set $i = i + 1$.
    end
    - Perform a classical crowding distance assignment to the representative set $\mathcal{I}$ using p as objective function.
    - Assign each solution x^i the same crowding distance as his corresponding representative $(x^i, \hat{\xi})$ has, for all $i \in \{1, 2, \ldots, l\}$.
  end
end

The representative crowding distance assignment has the advantage that the decision maker has the possibility to change the representative parameter by choice. Regarding an uncertain multiobjective optimization problem $\mathcal{P}_K(U)$ the decision maker therefor is enabled to choose a representative parameter ξ out of the uncertainty set U representing one of the possible scenarios to occur. This chosen scenario may be the undisturbed, the most important or the most likely scenario, if some probabilities are known.

Once again we have been working with the Java-based framework for multiobjective optimization with metaheuristics `jMetal` in order to implement the `SV-NSGA-II`.

3.3.2 Implementation in `jMetal`

In this section we will shortly describe how we managed to implement the `SV-NSGA-II` methaheuristic within the given `jMetal` framework [10]. Due to the fact, that all of the methaheuristics provided in `jMetal` run on determin-

istic multiobjective optimization problems, we had to review and edit fundamental instruments of the framework in order to generate an evolutionary algorithm able to run on multiobjective set-valued optimization problems. We will describe the basic components necessary to enable a user who is familiar with `jMetal` to understand the idea behind our construction and which are enabling him to construct further parametrized set-valued multiobjective optimization problems as well as methaheuristics able to run on this class of optimization problems. A complete list of all implemented classes and interfaces is provided in appendix E.
The basic module necessary to implement the `SV-NSGA-II`, was to generate a class representing a solution $x \in \mathcal{X}$ of a parametrized set-valued optimization problem $\mathcal{SP}^S_{\preccurlyeq^\star_K}$. Therefor we were working with the already existent `SolutionSet` class which contains a set of `Solution` objects. We use such a `SolutionSet` object in order to store a subset of the domain of the parametrized objective map $p : \mathcal{X} \times S \to \mathbb{R}^k$. I.e. the `SolutionSet` object representing the solution $x \in \mathbb{R}^n$, stores a set of elements $e := (x, \xi) \in \{x\} \times S$ of the domain of p, for a predefined or randomly chosen set of sample points $\xi \in S$ of the parameter set $S \subseteq \mathbb{R}^m$ and each such element e is stored in a `Solution` object. How many sample points $\xi \in S$ are chosen is up to the decision maker and is controllable via the input parameter $s \in \mathbb{N}$. Weather these sample points are uniformly picked for each solution or randomly sampled has to be defined in the implementation of the current regarded set-valued optimization problem $\mathcal{SP}^S_{\preccurlyeq^\star_K}$. In order to generate the population based methaheuristic `SV-NSGA-II`, we had to implement the `SolutionSetSet` class which stores a set of `SolutionSet` objects. Thus such a `SolutionSetSet` object represents a set of solutions $x \in \mathbb{R}^k$ of a parametrized set-valued optimization problem $\mathcal{SP}^S_{\preccurlyeq^\star_K}$.
Each such set-valued problem inherits from the class `SetProblem` which itself inherits from the `Problem` class already given within the `jMetal` framework. All set-valued problems have to define the methods `evaluate()` and `evaluateConstraints()`. Both methods receive a `Solution` object representing an element $e = (x, \xi) \in \mathbb{R}^n \times S$ of the domain of p for the given set-valued problem. The first method evaluates the parametrized objective function $p(x, \xi) \in F(x)$ and the second one determines the overall constraint violation of the solution $x \in \mathbb{R}^n$. Additional methods necessary to define are the `newSolutionSet()` method without input parameter and its overloaded version which receives a `Solution` object representing the element $\hat{e} = (\hat{x}, \hat{\xi}) \in \mathbb{R}^n \times S$ again. Both have to return a new `SolutionSet` object, containing the elements $(\bar{x}, \xi^i) \in \{\bar{x}\} \times S, i \in \{1, 2, \ldots, s\}$ where $s \in \mathbb{N}$ is the user-defined amount of sample points. The first method chooses $\bar{x} \in B$ within a for this problem predefined hyperbox $B \subset \mathbb{R}^n$ randomly and the

second method uses its input parameter value $\hat{x}$. For both methods the implementer of the set-valued problem has to define how the parameters ξ^i are sampled within the given parameter set $S \subseteq \mathbb{R}^m$. We will give an example of application for sampling parameters in the next section, where we are presenting the simulation results of the `SV-NSGA-II` on a parametrized multiobjective set-valued optimization problem.

3.3.3 Simulation Results

In order to test the `SV-NSGA-II` we have extended the `jMetal` framework by seven classes representing the seven pre-orders $\preccurlyeq_K^*$ such as the respective strict partial order $\prec_K^*$ induced by them, for all $* \in \{u, l, s, c, mn, mc, m\}$. These classes are listed and described in the appendix E.5. Each of them contains a constructor whose input parameter is a `DegreeDominanceComparator2` object, which represents the partial order $\preceq_{\hat{K}(\delta)}$ induced by the pointed polyhedral cone $\hat{K}(\delta) = K(A(\frac{\pi}{180}\delta))$ of example 1 with δ in arc degree instead of radians. We have tested the `SV-NSGA-II` on the following test problem introduced by J. Jahn in [23].

Example 10 (Example 4.2 in [23])
Given a proper closed convex and pointed cone $K \subset \mathbb{R}^k$ we are regarding the following unconstrained multiobjective set-valued optimization problem

$$\begin{aligned} \mathcal{SP}_{\preccurlyeq_K^*}: \ & \min{}_{\preccurlyeq_K^*} \left\{ (y_1, y_2)^\mathsf{T} \in \mathbb{R}^2 \,|\, (y_1 - 2x_1^2)^2 + (y_2 - 2x_2^2)^2 \leq (1 + x_2^2)^2 \right\} \\ & s.t. \ x = (x_1, x_2)^\mathsf{T} \in \mathbb{R}^2. \end{aligned}$$

In order to enable our implementation of the `SV-NSGA-II` to run on this test problem we have to regard its parametrized and constrained version.

Example 11
Given a proper closed convex and pointed cone $K \subset \mathbb{R}^k$ we are regarding the following constrained parametrized multiobjective set-valued optimization problem

$$\begin{aligned} \bar{\mathcal{SP}}^S_{\preccurlyeq_K^*}: \ & \min{}_{\preccurlyeq_K^*} \left\{ (p_1(x,\xi), p_2(x,\xi))^\mathsf{T} \in \mathbb{R}^2 \,|\, \xi = (\xi_1, \xi_2)^\mathsf{T} \in [0,1]^2 \right\} \\ & s.t. \ x = (x_1, x_2)^\mathsf{T} \in \mathcal{X}, \end{aligned}$$

where the parametrized objective function is given through

$$\begin{aligned} p_1(x,\xi) &:= \xi_2(1 + x_2^2)^2 \cos(\xi_1 2\pi) + 2x_1^2, \\ p_2(x,\xi) &:= \xi_2(1 + x_2^2)^2 \sin(\xi_1 2\pi) + 2x_2^2. \end{aligned}$$

And the decision space $\mathcal{X} := \left\{x \in \mathbb{R}^2 \,|\, g_i(x) \leq 0, i \in \{1,2,3,4,\}\right\}$ is defined through the following inequality constraints

$$\begin{aligned} g_1(x) &= -x_1 + a, & g_2(x) &= x_1 - b, \\ g_3(x) &= -x_2 + a, & g_4(x) &= x_2 - b. \end{aligned}$$

Whereas $a, b \in \mathbb{R}^2$ are user-defined parameters. Note that the set-valued objective function

$$F(x) := \left\{(p_1(x,\xi), p_2(x,\xi))^\mathsf{T} \in \mathbb{R}^2 \,|\, \xi = (\xi_1, \xi_2)^\mathsf{T} \in [0,1]^2\right\}$$

maps onto disks in $\mathbb{R}^2$ with variable radius $(1 + x_2^2)^2$ and center $(2x_1^2, 2x_2^2)^\mathsf{T}$ for each solution $(x_1, x_2)^\mathsf{T} \in \mathcal{X}$.

In `jMetal` it is necessary to restrict the decision space in each objective through an upper and a lower boundary through the instance variables `lowerLimit_` and `upperLimit_` of the `Problem` class, hence we could not regard the unrestricted optimization problem of example 10 from Jahn's paper [23]. Notice that this restriction can not be broken by any of the mutation and crossover operators used in line 5 of algorithm 5 as it limits the `encodings.variable` values to the defined boundaries. Notice as well, that this allows us to alleviate our restrictions on the optimization problems regarded in the sections on computational methods, as we are allowed to regard problems with a smaller domain, which is a subset of the hyperbox defined through the `lowerLimit_` and `upperLimit_` variables of the respective problem.

Jahn presented a derivative-free descent method in [23] which is able to search for one efficient solution of a multiobjective set-valued optimization problem $\mathcal{SP}_{\preccurlyeq_K^s}$ with non-empty and compact objective map images $F(x) \subset \mathbb{R}^k$ and convex sets $F(x) + K$ and $F(x) - K$ for all solutions $x \in \mathcal{X}$. His method fits into the class of pattern search methods. It starts with an user-defined solution $x^0 \in \mathbb{R}^n$ and develops a descent direction and step length in each iteration through clustering an equally distributed set of solutions around the current regarded one. The algorithm presented by Jahn then calculates the best descent direction as well as an appropriate step length, through comparison of these solutions. Therefor $4\bar{s}$ optimization problems have to be solved, whenever testing wether one solution dominates another one, as he replaces the dual cone K^* by a subset of $\bar{s} \in \mathbb{N}$ unit vectors. We on the other hand are not replacing the dual cone but are working with a discretization $\{p(x,\xi^1), \ldots, p(x,\xi^s)\}$ of the image $F(x)$ of each solution $x \in \mathbb{R}^n$. The amount of sample points $s \in \mathbb{N}$ defines the computational effort necessary to

test wether one solution $x \in \mathbb{R}^n$ dominates another solution $\bar{x} \in \mathbb{R}^n$. Let us have a look back on the binary relations introduced in section 3.1.1 and let us regard definition 3.5 of the lower set less order relation $\preccurlyeq_K^l$ as example. In order to test wether $F(x) \preccurlyeq_K^l F(\bar{x})$ holds true or not, in the worst case we will need to compare each of the sampled points of $F(x)$ with all of the sampled points of $F(\bar{x})$ w.r.t. $\preceq_K$. This requires $O(ks^2)$ comparisons, where k is the dimension of the objective space as usual. For the other binary relations presented in section 3.1.1 we gain the same number for the computational complexity or this number even increases. For example 11 the user defines the amount of sample points $s \in \mathbb{N}$ through two input parameters $\phi, r \in \mathbb{N}$ as described in algorithm 7.

Algorithm 7 : Sampling Process for Example 11

Input: Parameter $\phi, r \in \mathbb{N}$.
begin
- Set $i = \phi$, $j = r$ and $S_D = \{0 \in \mathbb{R}^2\}$.
  while $i \geq 1$ **do**
    while $j \geq 1$ **do**
      - Set $\xi_1 = \frac{i}{\phi}$, $\xi_2 = \frac{j}{r}$ and $\xi = (\xi_1, \xi_2)^\mathsf{T}$.
      - Add ξ to S_D.
      - Set $j = j - 1$.
    end
  - Set $i = i - 1$.
  end
end
Output: Discretization S_D of the parameter set S.

The discretization S_D of the parameter set S, which we gain as output of algorithm 7 contains $s := \phi r + 1$ sample points. Figure 16 depicts $F(x)$ in blue and its discretization $\{p(x, \xi^1), \ldots, p(x, \xi^s)\}$ in red, for the solution $x = 0 \in \mathbb{R}^2$ and input parameters $\phi = 20$ and $r = 5$ of algorithm 7.
We have tested the `SV-NSGA-II` on example 11 for each of the seven implemented pre-orders $\preccurlyeq_K^*$ with $* \in \{u, l, s, c, mn, mc, m\}$, where the inducing cone K equals the polyhedral cone $\hat{K}(\delta) = K(A(\frac{180}{\pi}\delta))$ of example 1 with $\delta \in [0°, 45°)$ in arc degree instead of radians. In these simulation runs we have been working with the adapted binary tournament selection described in section 3.3.1, the simulated binary crossover with distribution index 20,

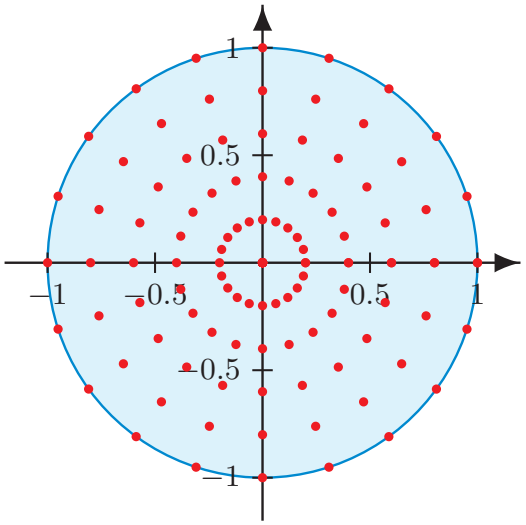


Figure 16: $F(0)$ and its discretization

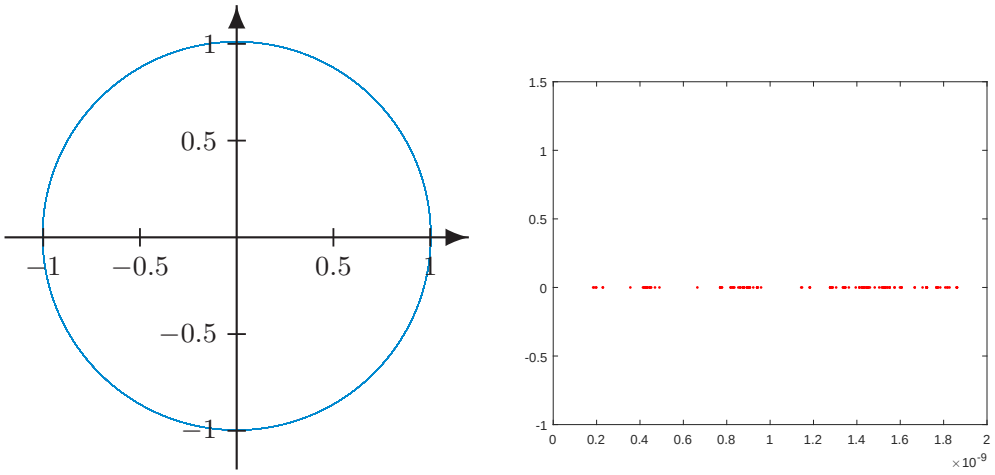
a crossover probability of 90% and the polynomial mutation operator with a probability of 50% and distribution index 20 (see e.g. [7] for further explanations). For reasons of clarity, additionally we have stated all input parameter settings used in our simulation runs in table 3.3.3. Here the parameters $a, b \in \mathbb{R}$ determine the decision space $\mathcal{X} = [a, b]^2$ as in example 11.

Table 1: Input parameters for the simulation runs on example 11

	δ	N	M	ϕ	r	ξ	$[a$	$b]$
$\preccurlyeq_K^u$	0	200	40000	30	10	$(0,0)^\mathsf{T}$	0	10
$\preccurlyeq_K^l$	0	300	60000	30	10	$(0.5,0.5)^\mathsf{T}$	0	10
$\preccurlyeq_K^s$	0	200	40000	30	10	$(0.5,0.5)^\mathsf{T}$	-4	4
$\preccurlyeq_K^c$	15	200	40000	20	5	$(0,0)^\mathsf{T}$	-4	4
$\preccurlyeq_K^{mn}$	30	100	20000	15	5	$(0,0)^\mathsf{T}$	-4	4
$\preccurlyeq_K^{mc}$	0	100	20000	15	5	$(0,0)^\mathsf{T}$	0	10
$\preccurlyeq_K^m$	10	100	20000	15	5	$(0,0)^\mathsf{T}$	0	25

Note that a definition of the decision space as a subset $\mathcal{X} = [0, b]^2, b \in \mathbb{R}_+$ of the first octant $\mathbb{R}^2_+$, does not restrict the objective space in such a manner that we could lose parts of the efficient front reachable with a decision space $\mathcal{X} = [-b, b]^2$. This results from the definition of the set-valued objective functions F of example 11, as $F(x_1, x_2) = F(-x_1, x_2) = F(x_1, -x_2) = F(-x_1, -x_2)$ holds true, for all $(x_1, x_2)^\mathsf{T} \in \mathbb{R}^2$.

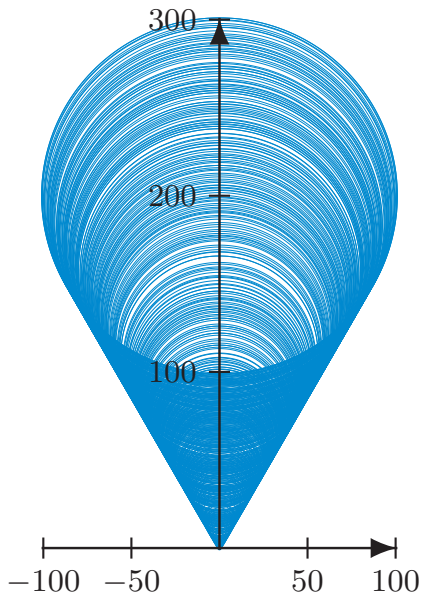
For the upper set less order relation $\preccurlyeq_K^u$ we chose the population size $N = 200$, the amount of iterations $M = 40000$, the input parameters of algorithm 7 defining the amount of samples $\phi = 30, r = 10$, the Edgeworth-Pareto cone $K = \mathbb{R}^2 = \hat{K}(0)$, the representative parameter $(0,0)^\mathsf{T} \in S$ as input of algorithm 6 and the square $[0,10]^2$ to randomly generate the first parent population P_1 in. Figure 17 illustrates the approximation of the efficient front $\mathcal{E}(\mathcal{SP}^S_{\preccurlyeq^u_{\mathbb{R}^2}})$ obtained by `SV-NSGA-II` on the left and the efficient set $\mathcal{S}_{\preccurlyeq^u_{\mathbb{R}^2}}$ obtained on the right. Here, the whole population evolved towards



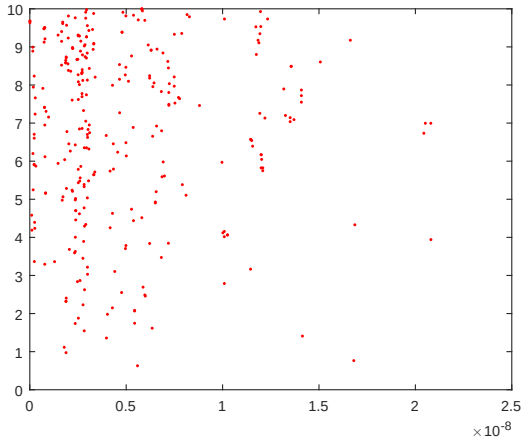
(a) Obtained efficient front $\mathcal{E}(\mathcal{SP}^S_{\preccurlyeq^u_{\mathbb{R}^2}})$ (b) Obtained efficient set $\mathcal{S}_{\preccurlyeq^u_{\mathbb{R}^2}}$

Figure 17: `SV-NSGA-II` on example 11 with input parameter $\preccurlyeq_K^u$

the origin in the decision space, whose image under the set-valued objective map F equals the unit disk. The `SV-NSGA-II` does not distinguish between the varying x_1 values visualized on the right of figure 17, due to the limited accuracy of a `double` value in `Java`.
Figure 18 depicts the output for the lower set less order relation $\preccurlyeq_K^l$ with population size $N = 300$, amount of iterations $M = 60000$, input parameters $\phi = 30, r = 10$ of algorithm 7, the Edgeworth-Pareto cone $K = \mathbb{R}^2$, the representative parameter $(0.5, 0.5)^\mathsf{T} \in S$ and the square $\mathcal{X} = [0,10]^2$ as decision space. Here the population evolved towards the x_2-axis and again does the limited accuracy of a `double` value within `Java` explain the varying x_1 values.

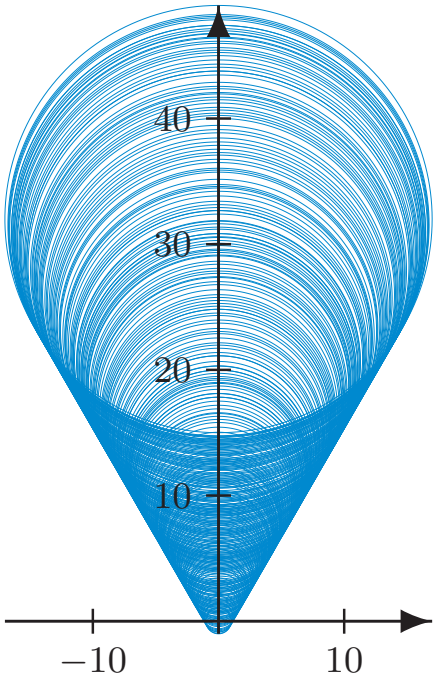


(a) Obtained efficient front $\mathcal{E}(\mathcal{SP}^S_{\preccurlyeq^l_{\mathbb{R}^2}})$

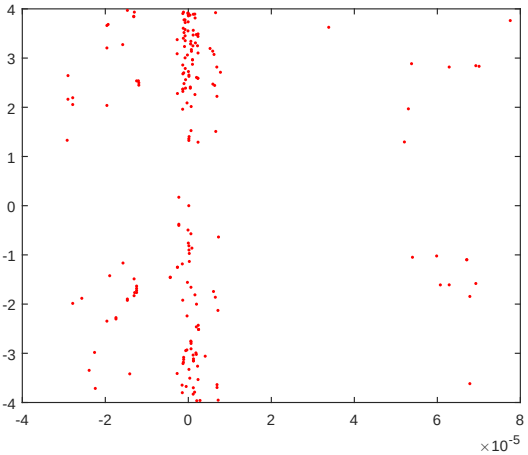


(b) Obtained efficient set $\mathcal{S}_{\preccurlyeq^l_{\mathbb{R}^2}}$

Figure 18: `SV-NSGA-II` on example 11 with input parameter $\preccurlyeq^l_K$



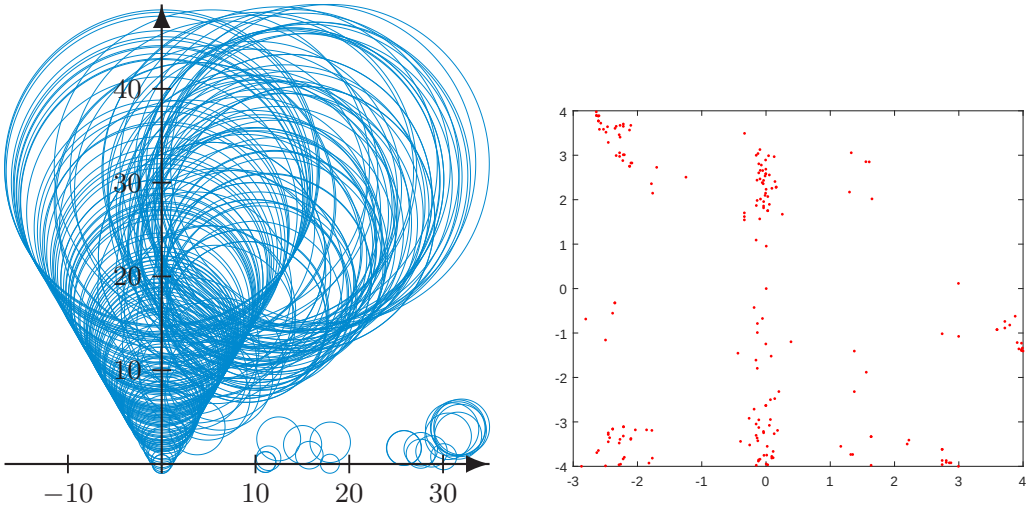
(a) Obtained efficient front $\mathcal{E}(\mathcal{SP}^S_{\preccurlyeq^s_{\mathbb{R}^2}})$



(b) Obtained efficient set $\mathcal{S}_{\preccurlyeq^s_{\mathbb{R}^2}}$

Figure 19: `SV-NSGA-II` on example 11 with input parameter $\preccurlyeq^s_K$

Figure 19 depicts the output for the set less order relation $\preccurlyeq_K^s$ with population size $N = 200$, amount of iterations $M = 40000$, input parameters $\phi = 30, r = 10$ of algorithm 7, the Edgeworth-Pareto cone $K = \mathbb{R}^2$, the representative parameter $(0.5, 0.5)^\mathsf{T} \in S$ and the square $\mathcal{X} = [-4, 4]^2$ as decision space. Here again it is clearly recognizable, that the population evolved towards the x_2-axis in the decision space, which was to be expected due to the definition 3.7 of the set less order relation $\preccurlyeq_K^s$ through the upper and lower set less order relations $\preccurlyeq_K^u$, $\preccurlyeq_K^l$.
For the certainly less order relation $\preccurlyeq_K^c$ we chose the population size $N = 200$, the amount of iterations $M = 40000$, the input parameters of algorithm 7 $\phi = 30, r = 10$, the cone $K = \hat{K}(15°)$ the representative parameter $(0, 0)^\mathsf{T} \in S$ and the square $\mathcal{X} = [-4, 4]^2$ as decision space. Figure 20 illustrates the efficient front $\mathcal{E}(\mathcal{SP}^S_{\preccurlyeq^c_{\hat{K}(15°)}})$ obtained by `SV-NSGA-II` on the left and the obtained efficient set $\mathcal{S}_{\preccurlyeq^s_{\hat{K}(15°)}}$ on the right.



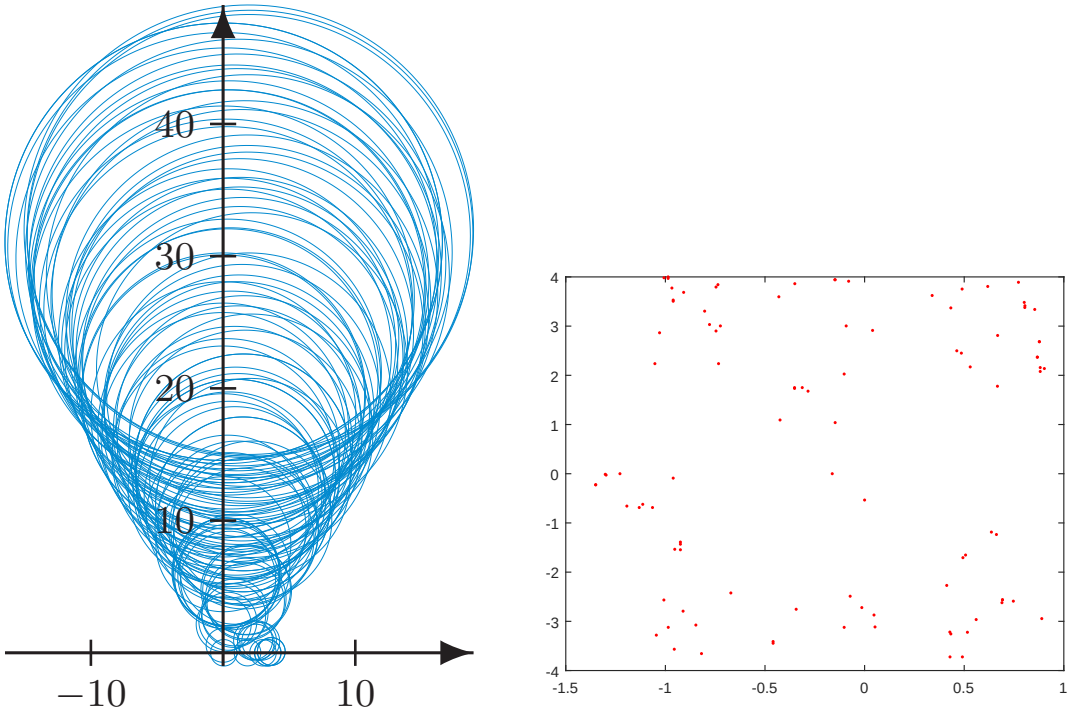
(a) Obtained efficient front $\mathcal{E}(\mathcal{SP}^S_{\preccurlyeq^c_{\hat{K}(15°)}})$

(b) Obtained efficient set $\mathcal{S}_{\preccurlyeq^c_{\hat{K}(15°)}}$

Figure 20: `SV-NSGA-II` on example 11 with input parameter $\preccurlyeq_K^c$

Here the obtained efficient front $\mathcal{E}(\mathcal{SP}^S_{\preccurlyeq^c_{\hat{K}(15°)}})$ and efficient sets $\mathcal{S}_{\preccurlyeq^c_{\hat{K}(15°)}}$, do look rather random at first sight. Though when having a closer look on the definition 3.8 of the certainly less order relation it becomes quite obvious that this strong ordering relation leads to a rather weak restriction for efficiency as it only filters out solutions which are dominated for every parameter by the same other solution. This explains the wider spread set of solutions we gained as output of this adjustment of parameters.

Figure 21 depicts the output for the minmax certainly nondominated order relation $\preccurlyeq_K^{mn}$ with population size $N = 100$, amount of iterations $M = 20000$, input parameters of algorithm 7 $\phi = 15, r = 5$, the pointed polyhedral cone $K = \hat{K}(30°)$, the representative parameter $(0,0)^\mathsf{T} \in S$ and the square $\mathcal{X} = [-4,4]^2$ as decision space.

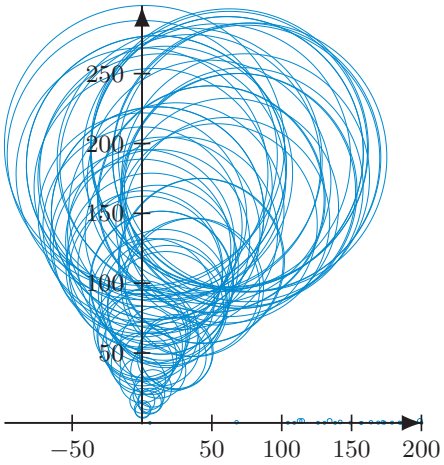


(a) Obtained efficient front $\mathcal{E}(\mathcal{SP}^S_{\preccurlyeq^{mn}_{\hat{K}(30°)}})$

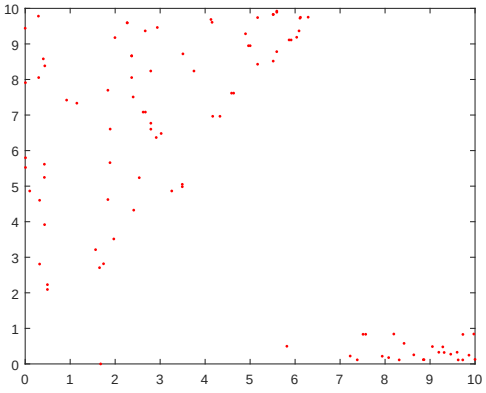
(b) Obtained efficient set $\mathcal{S}_{\preccurlyeq^{mn}_{\hat{K}(30°)}}$

Figure 21: `SV-NSGA-II` on example 11 with input parameter $\preccurlyeq_K^{mn}$

Here again an evolution of the population of solutions towards the x_2-axis in the decision space is recognizable. Note, that the efficient set $\mathcal{S}_{\preccurlyeq^{mn}_{\mathbb{R}^2}}$ is spread wider in the decision space as on the second and third adjustment of parameters. This is a result of the definition 3.11 of the minmax certainly nondominated order relation, as it is a quite strong ordering relation as well. For the minmax certainly set less order relation $\preccurlyeq_K^{mc}$ we chose the population size $N = 100$, the amount of iterations $M = 20000$, the input parameters of algorithm 7 $\phi = 15, r = 5$, the Edgeworth-Pareto cone $K = \mathbb{R}^2_+$, the representative parameter $(0,0)^\mathsf{T} \in S$ and the square $\mathcal{X} = [-4,4]^2$ as decision space. Figure 22 illustrates the efficient front $\mathcal{E}(\mathcal{SP}^S_{\preccurlyeq^{mc}_{\mathbb{R}^2}})$ obtained by `SV-NSGA-II` on the left and the obtained efficient set $\mathcal{S}_{\preccurlyeq^{mc}_{\mathbb{R}^2}}$ on the right.



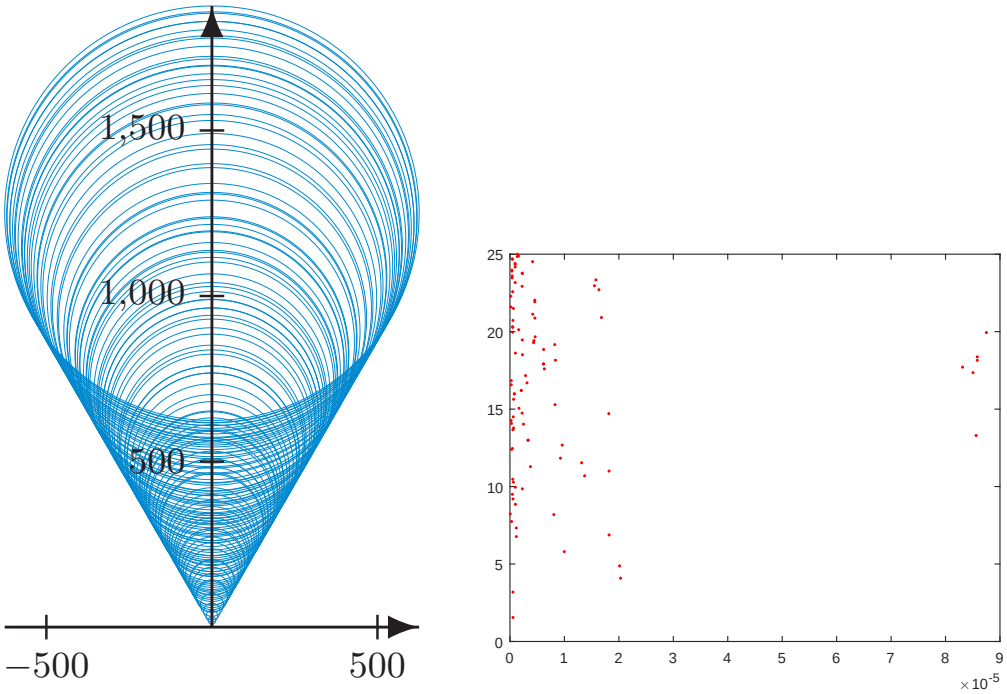
(a) Obtained efficient front $\mathcal{E}(\mathcal{SP}^S_{\preccurlyeq^{mc}_{\mathbb{R}^2}})$



(b) Obtained efficient set $\mathcal{S}_{\preccurlyeq^{mc}_{\mathbb{R}^2}}$

Figure 22: `SV-NSGA-II` on example 11 with input parameter $\preccurlyeq^{mc}_K$

Note that the efficient set $\mathcal{S}_{\preccurlyeq^{mc}_{\mathbb{R}^2}}$ obtained at this parameter setting as well contains solutions x^1, x^2 within ϵ-neighborhoods $B_\epsilon(x)$ of solutions $x = (x_1, 0)$ on the x_1-axis. This results from the definition 3.10, as $\min F(x^1) \preccurlyeq^c_{\mathbb{R}^2} \min F(x^2)$ does not hold true for two solutions out of this area.
Finally we have tested the `SV-NSGA-II` on the minmax less order relation $\preccurlyeq^m_K$, with population size $N = 100$, amount of iterations $M = 20000$, input parameters of algorithm 7 $\phi = 15, r = 5$, the pointed polyhedral cone $K = \hat{K}(10°)$, the representative parameter $(0, 0)^\mathsf{T} \in S$ and the square $\mathcal{X} = [-4, 4]^2$ as decision space. Figure 23 depicts the obtained efficient front $\mathcal{E}(\mathcal{SP}^S_{\preccurlyeq^m_{\hat{K}(10°)}})$ on the left and the obtained efficient set $\mathcal{S}_{\preccurlyeq^m_{\hat{K}(10°)}}$ on the right.



(a) Obtained efficient front $\mathcal{E}(\mathcal{SP}^S_{\preccurlyeq^m_{\tilde{K}(10°)}})$

(b) Obtained efficient set $\mathcal{S}_{\preccurlyeq^m_{\tilde{K}(10°)}}$

Figure 23: `SV-NSGA-II` on example 11 with input parameter $\preccurlyeq^m_K$

Again a strong development directed towards the x_2-axis is recognizable which is explained through the definition 3.9 of the regarded ordering relation $\preccurlyeq^m_K$.

3.4 Conclusions

In section 3 we have presented the concept of uncertain multiobjective optimization and pointed out strong connections to multiobjective set-valued optimization. We introduced seven ordering relations which lead to different declarations of what is considered to be a robust or efficient solution to an uncertain or set-valued optimization problem respectively. At last we have presented a first evolutionary algorithm able to run on parametrized multiobjective set-valued optimization problems or uncertain multiobjective optimization problems respectively and some numerical results in low dimensions depicting the variety of outputs possible. This opens a new field of application for evolutionary algorithms and presents a way into searching for the whole efficient front of set-valued or all robust efficient solutions to uncer-

tain optimization problems respectively. From here, further research may be done in many different directions. In practice, the `SV-NSGA-II` is very time-consuming. A deeper analysis of the presented binary relations is of interest in order to develop more efficient sampling and comparison methods to decrease the algorithms computational complexity. Also further selection and diversity mechanisms applicable to set-valued optimization problems could be an interesting point of research, as well as defining and inserting quality indicators. A deeper study of the influences of each of the various input parameters on the output of the `SV-NSGA-II` is desirable as well. More ordering relations on power sets as those we used in this thesis can be found in literature, for instance the possibly less order relation (see e.g. [24]) or the alternative set less order relation introduced in [18]. Those could be analyzed and integrated to the algorithm presented. Finally an adaptation of the various methaheuristics presented in literature that are different to `NSGA-II` (see e.g. [7]), in such a way that enables them to run on set-valued or uncertain optimization problems is desirable as well. The first step in this direction is presented through the extending implementations made for the `jMetal` framework as part of this thesis and its documentation presented in section 3.3.2 and E, allowing an easy access to future research developments towards this direction.

4 Closing Remarks

In total this thesis presented an introduction into the wide field of robustness in multiobjective optimization. In the first part a completely new notion of robustness was introduced, analyzed and a new methaheuristic was presented. The second part was covering the relatively new field of uncertain multiobjective optimization in combination with multiobjective set-valued optimization and presented a way for evolutionary algorithms to gain these research areas as new field of application.
Both parts leave open questions and possibilities for further research as discussed in section 2.2 and section 3.4. Furthermore a fusion of both areas of research is of high interest. The basic modules allowing this combinational study are presented through the theoretical as well as the practical part of this thesis. A big advantage for a computational application onto the field of evolutionary algorithms is given through the `Java` based implementations presented in this thesis, which are all included in the `jMetal` framework and hence are easy to combine with each other or different methaheuristics, operators, or other components contained in the framework.

References

[1] H.A. Abbass, L.T. Bui, M. Barlow & A. Bender, *Robustness Against the Decision-Maker's Attitude to Risk in Problems With Conflicting Objectives,* IEEE Transaction on evolutionary computation, Vol. 16, No. 1, 2012.

[2] S. Agaarwal, K. Deb, T. Meyarivan & A. Pratap, *A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II,* IEEE Transaction on evolutionary computation, Vol. 6, No. 2, 2002.

[3] H. Ansari & J. Yao., *Recent Developments in Vector Optimization,* Springer, ISBN 978-3-642-21113-3, 2012.

[4] J. Branke, *Creating Robust Solutions by Means of an Evolutionary Algorithm,* Parallel Problem Solving from Nature, p.119-128, LNCS 1498, Springer Verlag, 1998.

[5] A. Chiriaev & G.W. Walster, *Iverval Arithmetic Specification,* Technical Report, 1998.

[6] S. Das & P.N. Suganthan, *Differential evolution: a survey of the state-of-the-art,* IEEE Trans. Evol. Comput. 15, 4-31, 2011.

[7] K. Dep, *Multi-Objective Optimization using Evolutionary Algorithms,* Wiley, ISBN:0-471-87339-X, 2001.

[8] K. Dep & H. Gupta, *Introducing Robustness in Multi-Objective Optimization,* Massachusetts Institute of Technology, Evolutionary Computation 14(4): 463-494, 2006.

[9] J.J. Durillo & A.J. Nebro, *jMetal: A Java Framework for Multi-Objective Optimization,* Advances in Engineering Software 42, 760-771, 2011.

[10] J.J. Durillo & A.J. Nebro, *jMetal web site,* Last update January 29th 2015, Online at: URL: `http://jmetal.sourceforge.net/` (04.05.2015)

Edgeworth, F.Y.: Mathematical Psychics. Kegan Paul, London (1881)

[11] F.Y. Edgeworth, *Mathematical Psychics,* Kegan Paul, London, 1881.

[12] M. Ehrgott, J.Ide, & A. Schoebel, *Minmax Robustness for Multi-objective Optimization Problems,* European Journal of Operational Research, 2014.

[13] M. Ehrgott, *Multicriteria Optimization,* ISBN 3-540-21398-8, Springer, 2005.

[14] G. Eichfelder & J. Jahn, *Vector Optimization Problems and Their Solution Concepts,* Recent developments in vector optimization, Vector Optim., pages 1-27, Springer, Berlin, 2012.

[15] C. Feuersänger, *pgfplots - Create normal/logarithmic plots in two and three dimensions,* Version 1.12.1, Online at: URL: `https://ctan.org/pkg/pgfplots` (20.05.2015).

[16] A. Hilbert, *Mathematik (in German),* ISBN: 3-343-00248-8, VEB Fachbuchverlag Leipzig, 1987.

[17] J. Ide, *Concepts of Robustness for Uncertain Multi-Objective Optimization,* Dissertation at the GAUSS, 2014.

[18] J. Ide & E. Koebis, *Concepts of Efficiency based on Set Order Relations,* Technical report, Preprint-Reihe, Institut für Numerische und Angewandte Mathematik, Universitaet Goettingen, 2013.

[19] J. Ide, E. Koebis, D. Kuroiwa, A. Schoebel & C. Tammer, *The Relationship Between Multi-objective Robustness Concepts and Set-valued Optimization,* Fixed Point Theory and Applications, 2014.

[20] J. Ide, & A. Schoebel, *Robustness for Uncertain Multi-objective Optimization,* Technical report, Preprint-Reihe, Institut für Numerische und Angewandte Mathematik, Universität Göttingen, 2014.

[21] J. Ide, M. Tiedemann, S. Wesphal & F. Haiduk, *An Application of Deterministic and Robust Optimization in the Wood Cutting Industry,* Technical report, Preprint-Reihe, Institut für Numerische und Angewandte Mathematik, Universität Göttingen, 2013.

[22] J. Jahn, *Introduction to the Theory of Nonlinear Optimization,* Springer, ISBN: 978-3-540-49378-5, 2007.

[23] J. Jahn, *A Derivative-free Descent Method in Set Optimization,* Springer Science+Business Media, New York, 2014.

[24] J. Jahn & T.X.D. Ha, *New Order Relations in Set Optimization,* J. Optim. Theory Appl.,148(2):209-236, 2011.

[25] K. Jaenich, *Lineare Algebra(German),* Elfte Auflage, Springer, ISBN 978-3-540-75501-2, 2008.

[26] A. Khan, C. Tammer & F. Haiduk, C.Zalinescu *Set-valued Optimization,* Springer, ISBN 978-3-642-54264-0, 2015.

[27] L. Kaup, *Vorlesung über Torische Varietaeten(German),* lecture notes at the university of Konstanz, Konstanzer Schriften in Mathematik und Informatik Nr 130,ISSN 14030-35581931, Fassung vom Fruehjahr 2002.

[28] A. Kirsch, *Optimierungstheorie(German),* lecture notes at the KIT, Skript zur Vorlesung im Sommersemester 2005.

[29] O. Kramer, D.E. Ciaurri & S. Koziel, *Derivative-free optimization,* Springer, Computational Optimization, Methods and Algorithms, ed., 61-83, 2011.

[30] D. Kuroiwa, *Some Duality Theorems of Set-valued Optimization With Natural Criteria,* In T. Tanaka, editor, Proceedings of the international Conference on Nonlinear Analysis and Convex Analysis. World Scientific, 221-228, 1999.

[31] D. Kuroiwa, *Natural Criteria of Set-valued Optimization,* Manuscript at the Shimane University, Japan, 1998.

[32] D. Kuroiwa & G.M. Lee, *On Robust Multi-objective Optimization,* Vietnam Journal of Mathematics, 40, 305-317, 2012.

[33] H. Nakayama, Y. Sawaragi & T. Tanino *Theory of Multiobjective Optimization,* Mathematics in Science and Engineering, Volume 176, 1985.

[34] Z.G. Nishnianidze, *Fixed points of monotone multivalued operators,* Soobshch. Akad. Nauk Gruzin. SSR, 114(3):489-491, 1984.

[35] G.L. Nehmhauser & L.A. Wolsey, *Integer and Combinatorial Optimization,* ISBN: 0-471-82819-X, Wiley, 1988.

[36] P. K. Shukla, M. A. Braun & H. Schmeck *Theory and Algorithms for Finding Knees,* Springer, Evolutionary Multi-Criteron Optimization, Lecture Notes in Computer Science Volume 7811, pp 156-170, 2013.

[37] A. Soyster, *Convex Programming with Set-inclusive Constraints and Applications to Inexact Linear Programming,* Operations Research, 21:1154-1157, 1973.

[38] W. Stadler, *Multicriteria Optimization in Engineering and in the Sciences,* Plenum Press, New York, 1988.

[39] V. Pareto, *Manuale di economia politica,* Societa Editrice, Libraria, Milano, Italy, 1906, English translation: V. Pareto, Manual of Political Economy, translated by A.S. Schwier, M. Augustus, Kelley Publishers, New York, 1971.

[40] R. Takahashi, K. Deb, E. Wanner & S. Greco *Evolutionary Multi-Criterion Optimization,* 6th International Conference, EMO 2011, Ouro Preto, Brazil, Springer, 2011.

[41] K.C. Tan, E.F. Khor & T.H. Lee, *Multiobjective Evolutionary Algorithms and Applications,* Springer, 2005.

[42] T. Tantau, *pgf - Create PostScript and PDF graphics in TeX,* Version 3.0.0 2013-12-20, Online at: URL: `https://www.ctan.org/pkg/pgf` (20.05.2015).

[43] S. Tsutsui & A. Ghosh, *Genetic algorithms with a robust solution searching scheme,* IEEE Transactions on Evolutionary Computation, p. 201-219, 1997.

[44] M.M. Wiecek, *Advances in Cone-Based Preference Modeling for Decision Making with Multiple Criteria,* Decision Making in Manufacturing and Services, Vol. 1, No. 1-2, pp. 153-173, 2007.

[45] R.C. Young, *The Algebra of Many-valued Quantities,* Math. Ann., 104(1):260-290, 1931.

[46] E. Zitzler, K. Deb, L. Thieler, *Comparison of multiobjective evolutionary algorithms: Empirical results,* IEEE Trans. on Evol. Computation 8, 173-195, 2000.

A Consistence of the Definition of an Edge

Throughout this appendix $K := K(A) \subseteq \mathbb{R}^k$ will be a polyhedral cone, with A being an arbitrary real $l \times k$-matrix having the row vectors $a_i \in \mathbb{R}^k$, $i \in I := \{1, 2, \ldots, l\}$. To proof the consistence of our edge definition of a k-edged cone with the one usually used in literature, we'll need the following definitions and results (see e.g. [27], [35]).

Definition A.1
For a polyhedral cone K we define the following two sets of indexes

$$I^{=} := \{\ i \in I \,|\, \langle y, a_i \rangle = 0,\ \forall y \in K\}$$
$$I^{\geq} := \{\ i \in I \,|\, \exists y \in K : \langle y, a_i \rangle > 0\}\,.$$

And $A^{=}$ is the $|I^{=}| \times k$-matrix which has the row vectors a_i for all $i \in I^{=}$ and $A^{\geq}$ the matrix having the to $I^{\geq}$ corresponding rows of A respectively.

Definition A.2
The dual cone *K^* of K is defined to be*

$$K^* := \{y \in \mathbb{R}^k \,|\, \langle y, k \rangle \geq 0,\ \forall k \in K\}.$$

Theorem A.3
The following three statements hold true

(i) $(K^*)^* = K$,

(ii) K *is pointed* $\iff \dim(K^*) = k \iff rg(A) = k$,

(iii) $\dim(K) + rg(A^{=}) = k$.

Proof: See [27] corollary 1.8 and note 1.13 for (i) and corollary 1.21 for (ii), whereas the last equality follows directly from the next corollary. For (iii) see proposition 2.4 of [35]. ■

Corollary A.4
The following holds true

$$K(A)^* = cone\Big(\big\{a_i \,|\, i \in \{1, 2, \ldots, l\}\ \big\}\Big).$$

Proof: Let $k \in K(A)$

$$\iff \langle k, a_i \rangle \geq 0,\ \ \forall i \in \{1, 2, \ldots, l\}$$

$$
\begin{aligned}
&\iff \sum_{i=0}^{l} \lambda_i \langle k, a_i \rangle \geq 0, \;\; \forall \lambda_i \geq 0, \forall i \in \{1, 2, \ldots, l\} \\
&\iff \langle k, \sum_{i=0}^{l} \lambda_i a_i \rangle \geq 0, \;\; \forall \lambda_i \geq 0, \forall i \in \{1, 2, \ldots, l\} \\
&\iff k \in cone\Big(\big\{ a_i \,|\, i \in \{1, 2, \ldots, l\} \big\} \Big)^*.
\end{aligned}
$$

Thus $K(A) = cone\Big(\big\{ a_i \,|\, i \in \{1, 2, \ldots, l\} \big\} \Big)^*$ holds true, which with theorem A.3 (i) is equivalent to the assertion. ■

Definition A.5
A subset F of K is called a face *of K iff there exists a vector $v \in K^*$ such that $F = K \cap v^\perp$, where $v^\perp := \{y \in \mathbb{R}^k \,|\, \langle y, v \rangle = 0\}$ is the hyperplane induced by v.*

Definition A.6
A face F of K is called an edge *iff F is one-dimensional.*

Now we can prove that our definition of an edge of a k-edged cone in $\mathbb{R}^k$ is equivalent to definition A.6.

Lemma A.7
Let $K(A)$ be a k-edged cone in $\mathbb{R}^k$ induced by the matrix $A \in \mathbb{R}^{k \times k}$. I.e. let $a_i \in \mathbb{R}^k$ be the i-th row vector of $A \in \mathbb{R}^{k \times k}$, for all $i \in I := \{1, 2, \ldots, k\}$ and let $rg(A) = k$ hold true. Then definition 1.10 of an edge of $K(A)$ is equivalent to definition A.6.

Proof: Let F be a face of $K(A)$, that's equivalent to

$$
\begin{aligned}
&\exists a \in K(A)^* : \; F = K(A) \cap a^\perp \\
\overset{A.4}{\iff} \; &\forall i \in I \, \exists \lambda_i \geq 0 : \; F = K(A) \cap \left\{ y \in \mathbb{R}^k \,|\, \left\langle y, \sum_{i=1}^{k} \lambda_i a_i \right\rangle = 0 \right\} \\
\iff \; &\forall i \in I \, \exists \lambda_i \geq 0 : \; F = K(A) \cap \left\{ y \in \mathbb{R}^k \,|\, \sum_{i=1}^{k} \lambda_i \langle y, a_i \rangle = 0 \right\} \\
\iff \; &\forall i \in I \, \exists \lambda_i \geq 0 : \; F = \left\{ y \in \mathbb{R}^k \,|\, \sum_{i=1}^{k} \lambda_i \langle y, a_i \rangle = 0, \; \langle y, a_i \rangle \geq 0 \, \forall i \in I \right\}.
\end{aligned} \tag{17}
$$

Now let us regard the set of active indexes $I^{>} := \{\, i \in I \,|\, \lambda_i > 0\}$. If $I^{>} = \emptyset$ we gain $F = K(A)$, thus let's exclude this case in the following. Notice that F is a polyhedral cone again with $I^{=} = I^{>}$ and $I^{\geq} = I \setminus I^{>}$. Thus statement (17) is equivalent to

$$F = \left\{ y \in \mathbb{R}^k \,|\, \langle y, a_i \rangle = 0,\, \forall i \in I^{=}, \langle y, a_j \rangle \geq 0 \,\forall j \in I^{\geq} \right\} \overset{1.8}{\iff}$$

$$F = \left(\bigcap_{i \in I^{=}} P_i(A^{=}) \right) \cap \left(\bigcap_{j \in I^{\leq}} H(A^{\geq}) \right). \tag{18}$$

Since $rg(A) = k$ holds true, with $A \in \mathbb{R}^{k \times k}$ it follows that $rg(A^{=}) = |I^{=}|$. By theorem A.3 (iii) it follows $\dim(F) = n - |I^{=}|$. Thus $\dim(F) = 1$ iff $|I^{=}| = (k-1)$ which, with (18), is equivalent to

$$F = \left(\bigcap_{\substack{j=1, \\ j \neq i}}^{k} P_j(A) \right) \cap H_i(A),$$

for one $i \in I$.

■

B Proof of Example 1

Proof: Within this proof let $I := \{1, 2, \ldots, k\}$,

$$\mathsf{x} := \frac{\tan(\delta)}{\sqrt{k-1} - (k-2)\tan(\delta)} \tag{19}$$

and for an arbitrary square matrix $A \in \mathbb{R}^{k \times k}$ with row vectors $a_i, i \in I$ the matrix $A_i := (a_1, a_2, \ldots, a_{i-1}, a_{i+1}, \ldots, a_k)^{\mathsf{T}}$ will be the $(k-1) \times k$-matrix, which has identical row vectors as A except for leaving a_i out.

Let $\delta = 0$ hold true, then it follows that $A(\delta)$ equals the identity matrix I^k in $\mathbb{R}^k$. Thus with example 1.12 we gain that $K(A(0)) = \mathbb{R}^k_+$. For $A(0) = I^k$ the assumptions of definition 1.10 obviously hold true and with i_j being the j-th row vector of the unit matrix I^k it follows that

$$E_i(A(0)) = \left\{ y \in \mathbb{R}^k \,|\, I^k_i y = 0 \right\} \cap \mathbb{R}^k_+ = \{\lambda i_i \,|\, \lambda \geq 0\} = \{\lambda e^i(0) \,|\, \lambda \geq 0\}.$$

Now let $\delta > 0$, $\delta \in D$ hold true. First we will prove that for $A(\delta) \in \mathbb{R}^{k\times k}$ the assumptions of definition 1.10 are fulfilled in this case as well. I.e. we have to show that $rg(A(\delta)) = k$ holds true. Therefore let us transform the matrix $A(\delta)$ via row and column operations in the following way.

$$A(\delta) = \begin{pmatrix} 1 & \mathsf{x} & \mathsf{x} & \cdots & \mathsf{x} \\ \mathsf{x} & 1 & \mathsf{x} & & \vdots \\ \mathsf{x} & \mathsf{x} & \ddots & \ddots & \\ \vdots & & \ddots & & \mathsf{x} \\ \mathsf{x} & \cdots & & \mathsf{x} & 1 \end{pmatrix} \rightsquigarrow \begin{pmatrix} 1 & \mathsf{x} & \mathsf{x} & \cdots & \mathsf{x} \\ \mathsf{x}-1 & 1-\mathsf{x} & 0 & \cdots & 0 \\ \mathsf{x}-1 & 0 & 1-\mathsf{x} & \ddots & \vdots \\ \vdots & \vdots & \ddots & \ddots & 0 \\ \mathsf{x}-1 & 0 & \cdots & 0 & 1-\mathsf{x} \end{pmatrix}$$

$$\rightsquigarrow \begin{pmatrix} 1+(k-1)\mathsf{x} & \mathsf{x} & \mathsf{x} & \cdots & \mathsf{x} \\ 0 & 1-\mathsf{x} & 0 & \cdots & 0 \\ 0 & 0 & \ddots & \ddots & \vdots \\ \vdots & & \ddots & & 0 \\ 0 & \cdots & & 0 & 1-\mathsf{x} \end{pmatrix} =: B(\delta)$$

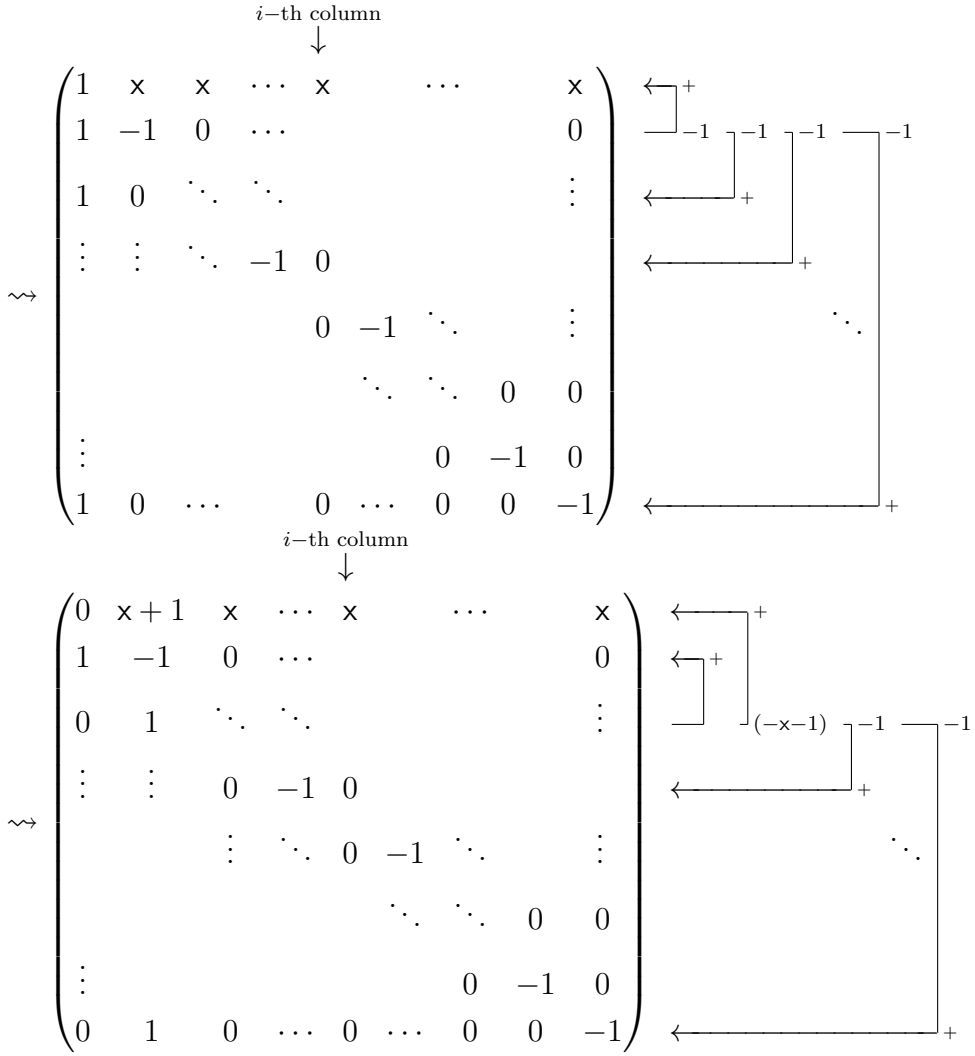
Thus $rg(A(\delta)) = rg(B(\delta)) = k$ iff $1 + (k-1)\mathsf{x} \neq 0$ and $1 - \mathsf{x} \neq 0$ holds true. By substituting x back, that is equivalent to $\delta \neq \arctan(-\sqrt{k-1})$ and $\delta \neq \arctan(\frac{1}{\sqrt{k-1}})$, which is fulfilled for $\delta \in D$.
Thus the assumptions of definition 1.10 hold true and we can calculate the edges $E_i(A(\delta)), i \in I$ of $K(A(\delta))$ as follows.

$$E_i(A(\delta)) = \left(\bigcap_{\substack{j=1,\\ j\neq i}}^{k} P_j(A(\delta)) \right) \cap H_i(A(\delta)) = \left\{ y \in \mathbb{R}^k \mid A(\delta)_i y = 0, \langle a_i, y \rangle \geq 0 \right\}. \tag{20}$$

In order to solve the system of linear equations determining the last set above, we will use the Gaussian elimination.

$$A(\delta)_i = \overset{i-\text{th column}}{\downarrow}\begin{pmatrix} 1 & \mathsf{x} & \mathsf{x} & \cdots & \mathsf{x} & & \cdots & & \mathsf{x} \\ \mathsf{x} & 1 & \mathsf{x} & & & & & & \vdots \\ \mathsf{x} & \mathsf{x} & \ddots & \ddots & & & & & \\ \vdots & & \ddots & 1 & \mathsf{x} & & & & \\ & & & \mathsf{x} & 1 & \ddots & & \vdots & \\ & & & & \ddots & \ddots & \mathsf{x} & \mathsf{x} & \\ \vdots & & & & & \mathsf{x} & 1 & \mathsf{x} & \\ \mathsf{x} & & \cdots & & \mathsf{x} & \cdots & \mathsf{x} & \mathsf{x} & 1 \end{pmatrix}$$

$$\rightsquigarrow \overset{i-\text{th column}}{\downarrow}\begin{pmatrix} 1 & \mathsf{x} & \mathsf{x} & \cdots & \mathsf{x} & & \cdots & & \mathsf{x} \\ \mathsf{x}-1 & 1-\mathsf{x} & 0 & \cdots & & & & & 0 \\ \mathsf{x}-1 & 0 & \ddots & \ddots & & & & & \vdots \\ \vdots & \vdots & \ddots & 1-\mathsf{x} & 0 & & & & \\ & & & & 0 & 1-\mathsf{x} & \ddots & & \vdots \\ & & & & & \ddots & \ddots & 0 & 0 \\ \vdots & & & & & & 0 & 1-\mathsf{x} & 0 \\ \mathsf{x}-1 & 0 & \cdots & & 0 & \cdots & 0 & 0 & 1-\mathsf{x} \end{pmatrix} \begin{matrix} \\ \mid \cdot \frac{1}{\mathsf{x}-1} \\ \mid \cdot \frac{1}{\mathsf{x}-1} \\ \vdots \\ \\ \\ \\ \mid \cdot \frac{1}{\mathsf{x}-1} \end{matrix}$$


i−th column
i−th column
(−x−1)

$$
\begin{array}{c}
\overset{i-\text{th column}}{\downarrow} \\
\rightsquigarrow \begin{pmatrix}
0 & 0 & 2x+1 & \mathsf{x} & \cdots & & & & & \mathsf{x} \\
1 & 0 & -1 & 0 & \cdots & & & & & 0 \\
0 & 1 & -1 & 0 & & & & & & \\
\vdots & 0 & 1 & -1 & \ddots & & & & & \vdots \\
 & \vdots & 1 & 0 & \ddots & & & & & \\
 & & \vdots & \vdots & \ddots & -1 & 0 & & & \\
 & & & & & & 0 & -1 & & \\
 & & & & & & & & \ddots & \ddots & 0 \\
 & & & & & & & & \ddots & -1 & 0 \\
0 & 0 & 1 & 0 & & \cdots & & & 0 & 0 & -1
\end{pmatrix}
\end{array}
$$

Now lets repeat this procedure, i.e. for all $l \in \{4, 5, \ldots, k-1\}$ add in ascending order the l-th row to the m-th row for all $m \in \{2, 3, \ldots, l-1\}$, add $(-(l-2)\mathsf{x}-1)$ times the l-th row to the first row and subtract the l-th row from the p-th row for all $p \in \{l+1, l+2, \ldots, k-1\}$. Then we gain the following matrix.

$$
\begin{array}{c}
\overset{i-\text{th column}}{\downarrow} \\
\begin{pmatrix}
0 & 0 & \cdots & 0 & \mathsf{x} & 0 & \cdots & & (k-2)\mathsf{x}+1 \\
1 & 0 & & \vdots & 0 & & & & -1 \\
0 & 1 & \ddots & & & & & & \vdots \\
0 & 0 & \ddots & & & & & & \\
 & & & 1 & 0 & & & & \vdots \\
\vdots & & & & 0 & 1 & \ddots & 0 & \\
 & & & & & & \ddots & 0 & -1 \\
0 & 0 & \cdots & & 0 & \cdots & 0 & 1 & -1
\end{pmatrix} \quad \left| \cdot \tfrac{1}{\mathsf{x}} \right.
\end{array}
$$

$$\overset{i-\text{th column}}{\overset{\downarrow}{}}$$

$$\rightsquigarrow \begin{pmatrix} 0 & 0 & \cdots & 0 & 1 & 0 & \cdots & & (k-2)+\frac{1}{\mathsf{x}} \\ 1 & 0 & & \vdots & 0 & & & & -1 \\ 0 & 1 & \ddots & & & & & & \vdots \\ 0 & 0 & \ddots & & & & & & \\ & & & 1 & 0 & & & & \vdots \\ \vdots & & & & 0 & 1 & \ddots & 0 & \\ & & & & & & \ddots & 0 & -1 \\ 0 & 0 & \cdots & & 0 & \cdots & 0 & 1 & -1 \end{pmatrix} =: \bar{A}(\delta)_i.$$

Thus follows for $y = (y_1, y_2, \ldots, y_k)^\mathsf{T} \in \mathbb{R}^k$, that

$$A(\delta)_i y = 0 \iff \bar{A}(\delta)_i y = 0. \tag{21}$$

Now we are substituting x back again. For $i \neq k$, (21) is equivalent to

$$\begin{aligned} &y_j = y_k \quad \forall j \in \{1, 2 \ldots, k-1\} \text{ with } j \neq i, \\ &y_i + (k-2+\frac{\sqrt{k-1}-(k-2)\tan(\delta)}{\tan(\delta)})y_k = 0 \iff y_k = -\frac{\tan(\delta)}{\sqrt{k-1}} y_i. \end{aligned}$$

For $i = k$, (21) is equivalent to

$$\begin{aligned} y_j = y_{k-1} \, \forall j \in \{1, 2, \ldots, k-2\}, \\ y_{k-1} = -\frac{\tan(\delta)}{\sqrt{k-1}} y_k. \end{aligned}$$

Together we gain

$$y_j = -\frac{\tan(\delta)}{\sqrt{k-1}} y_i \;\; \forall j \in I, j \neq i. \tag{22}$$

In combination with (20) it follows

$$\begin{aligned} & y \in E_i(K(A(\delta))) \\ \iff & y_i + \sum_{\substack{j=1, \\ j \neq i}}^{k} \left(\left(\frac{\tan(\delta)}{\sqrt{k-1}-(k-2)\tan(\delta)} \right) \left(-\frac{\tan(\delta)}{\sqrt{k-1}} \right) y_i \right) \geq 0 \end{aligned}$$

$$\iff \frac{-\tan^2(\delta)\sqrt{k-1}+\sqrt{k-1}-(k-2)\tan(\delta)}{\sqrt{k-1}-(k-2)\tan(\delta)}y_i \geq 0. \tag{23}$$

Obviously the denominator of the left fraction in the equation above is positive for $k = 2$. Since $\tan(\cdot)$ is a strictly increasing function on D,

$$\delta \in D \iff \tan(\delta) \in \left[0, \frac{1}{\sqrt{k-1}}\right) \tag{24}$$

and

$$\frac{1}{\sqrt{k-1}} < \frac{\sqrt{k-1}}{k-2} \iff 1 < \frac{k-1}{k-2}$$

holds true for all $k \geq 3$, follows that the denominator is positive for $k \geq 3$ as well. Thus (23) is equivalent to

$$\begin{aligned} & (-\sqrt{k-1}\tan^2(\delta) - (k-2)\tan(\delta) + \sqrt{k-1})y_i \geq 0 \\ \iff & \left(\tan(\delta)+\sqrt{k-1}\right)\left(-\sqrt{k-1}\tan(\delta)+1\right)y_i \geq 0 \\ \overset{k\geq 2}{\iff} & \left(\tan(\delta)+\sqrt{k-1}\right)\left(\tan(\delta)-\frac{1}{\sqrt{k-1}}\right)y_i \leq 0. \end{aligned} \tag{25}$$

Due to (24) it follows that $\tan(\delta)+\sqrt{k-1} \geq 0$ and $\tan(\delta) - \frac{1}{\sqrt{k-1}} \leq 0$ thus (25) is equivalent to $y_i \geq 0$. Hence together with (22), equation (5) of our example is proven.

■

C Proof of Example 3

Proof:
Equation (9) is a direct consequence of theorem 1.13, whose assertion is proven to be fulfilled through the proof of example 1.

Since $A(0) = I^k$ holds true, with example 1.12 we gain that $\bar{c}(\mathbb{R}^k_+, 0) = \mathbb{R}^k_+$ and hence point (i) of definition 2.2 is fulfilled as well. What is left to prove is that point (ii) of definition 2.2 holds true for our example. Remember, that equation (5) of example 1 is fulfilled, i.e. for all $\delta \in D$ holds

$$E_i(A(\delta)) = \left\{\lambda e^i(\delta) \,|\, \lambda \geq 0\right\}, \text{ with } e^i(\delta) := \begin{pmatrix} e^i_1(\delta) \\ e^i_2(\delta) \\ \vdots \\ e^i_k(\delta) \end{pmatrix}$$

$$\text{and } e^i_j(\delta) := \left\{ \begin{array}{ll} -\frac{\tan(\delta)}{\sqrt{k-1}} & \text{if } j \neq i, \\ 1 & \text{if } j = i \end{array} \right\}, \ \forall i,j \in \{1,2,\ldots,k\}.$$

In the following let $\delta_1, \delta_2 \in D$ with $\delta_1 < \delta_2$ hold true, then follows

$$\begin{aligned} & \bar{c}(K,\delta_1) \subset \bar{c}(K,\delta_2) \\ \overset{(9)}{\iff} \ & cone\Big(\big\{E_i(A(\delta_1)) \,|\, i \in I\big\}\Big) \subset cone\Big(\big\{E_i(A(\delta_2)) \,|\, i \in I\big\}\Big) \qquad (26) \\ \overset{1,7,(5)}{\iff} \ & \forall i,j \in I \, \forall \lambda_i \geq 0 \, \exists \mu_{ji} \geq 0 : \ \sum_{i=1}^{k} \lambda_i e^i(\delta_1) = \sum_{j=1}^{k} \mu_{ji} e^j(\delta_2). \qquad (27) \end{aligned}$$

By this obviously follows

$$\forall i,j \in I \, \exists \xi_{ji} \geq 0 : \ \ e^i(\delta_1) = \sum_{j=1}^{k} \xi_{ji} e^j(\delta_2) \iff \qquad (28)$$

$$\forall i,j \in I \forall \lambda_i \geq 0 \exists \xi_{ji} \geq 0 : \sum_{i=1}^{k} \lambda_i e^i(\delta_1) = \sum_{i=1}^{k} \lambda_i \sum_{j=1}^{k} \xi_{ji} e^j(\delta_2) = \sum_{j=1}^{k} \left(\sum_{i=1}^{k} \lambda_i \xi_{ji} \right) e^j(\delta_2).$$

By setting $\sum_{i=1}^{k} \lambda_i \xi_{ji} = \mu_i, \forall i \in I$ we gain (27) again and thus (26) is equivalent to $\big\{e^i(\delta_1) \,|\, i \in I\big\} \subset cone\Big(\big\{E_i(A(\delta_2)) \,|\, i \in I\big\}\Big)$ and whats left to prove is that (28) holds true.
To prove that that this is the case we will first show that (28) holds true for $\delta_1 = 0$. Therefor we will define $z(\delta) := -\frac{tan(\delta)}{\sqrt{k-1}}$ first, notice that

$$\delta \in D \iff z(\delta) \in \left(-\frac{1}{k-1}, 0 \right] \qquad (29)$$

holds true. Now we define for all $i,j \in I$

$$\xi_{ji}(\delta) := \left\{ \begin{array}{ll} \xi_{*i}(\delta) := \frac{-z(\delta)}{(1-z(\delta))((k-1)z(\delta)+1)} & \text{if } i \neq j, \\ \\ \frac{(k-2)z(\delta)+1}{(1-z(\delta))((k-1)z(\delta)+1)} & \text{otherwise.} \end{array} \right.$$

Then it follows with (29) that

$$-z(\delta) \in \left[0, \frac{1}{k-1} \right),$$

$$(1 - z(\delta)) \in \left[1, 1 + \frac{1}{k-1}\right),$$
$$((k-1)z(\delta) + 1) \in (0, 1]$$

is the case for all $\delta \in D$, hence $\xi_{*i}(\delta) \geq 0$ for all $i \in I$.
Since $(k-2)z(\delta) + 1 = ((k-1)z(\delta) + 1) + (-z(\delta))$ holds true, we gain

$$\xi_{ji}(\delta) \geq 0 \quad \text{for all } i, j \in I \text{ and } \delta \in D. \tag{30}$$

Now let us define $u^i(\delta) = (u_1^i(\delta), u_2^i(\delta), \ldots, u_k^i(\delta))^\mathsf{T} \in \mathbb{R}^k$, for $i \in I$ as

$$u^i(\delta) := \sum_{j=1}^{k} \xi_{ji}(\delta) e^j(\delta).$$

Due to $e_l^j(\delta) = z(\delta)$ holding true for all $\delta \in D$ and $l \in I$ with $l \neq j$, we gain

$$\begin{aligned} u_r^i(\delta) &= (k-2)\xi_{*i}(\delta)z(\delta) + \xi_{*i}(\delta) + \xi_{ii}(\delta)z(\delta) \\ &= \frac{-(k-2)z(\delta)^2 - z(\delta) + ((k-2)z(\delta) + 1)z(\delta)}{(1 - z(\delta))((k-1)z(\delta) + 1)} = 0 \end{aligned}$$

to be the case for all $\delta \in D$ and $r \in I$ with $r \neq i$. With

$$u_i^i(\delta) = (k-1)\xi_{*i}(\delta)z(\delta) + \xi_{ii}(\delta) = \frac{-(k-1)z(\delta)^2 + (k-2)z(\delta) + 1}{(1 - z(\delta))((k-1)z(\delta) + 1)} = 1$$

it follows that

$$e^i(\delta_1) = e^i(0) = i_i = u^i(\delta) = \sum_{j=1}^{k} \xi_{ji}(\delta) e^j(\delta)$$

is true, for all $\delta \in D$ and $i \in I$, where i_i is the i-th unit vector in $\mathbb{R}^k$. Hence (28) is proven for $\delta_1 = 0$. Now let $0 < \delta_1 < \delta_2$ hold true, then we have

$$e^i(\delta_1) = e^i(\delta_2) + \sum_{\substack{l=1,\\ l\neq i}}^{k} \Big(z(\delta_1) - z(\delta_2)\Big) i_l = e^i(\delta_2) + \sum_{\substack{l=1,\\ l\neq i}}^{k} \Big(z(\delta_1) - z(\delta_2)\Big) \sum_{j=1}^{k} \xi_{jl}(\delta_2) e^j(\delta_2)$$

$$= \underbrace{\left(\sum_{\substack{l=1,\\ l\neq i}}^{k} \Big(z(\delta_1) - z(\delta_2)\Big) \xi_{il}(\delta_2)\right)}_{\geq 0} e^i(\delta_2) + \sum_{\substack{j=1,\\ j\neq i}}^{k} \underbrace{\left(\sum_{\substack{l=1,\\ l\neq i}}^{k} \Big(z(\delta_1) - z(\delta_2)\Big) \xi_{jl}(\delta_2)\right)}_{\geq 0} e^j(\delta_2),$$

where the scalars in the last linear combination are nonnegative because of $\delta_1 < \delta_2$, the definition of $z(\delta)$ and (30). Thus (28) holds true for all $\delta_1, \delta_2 \in D$ with $\delta_1 < \delta_2$, which proves that $\bar{c}$ is a proper ascending cone mapping.

■

D Proof that Algorithm 3 is Well-Defined

In order to prove that the removal of already dominated solutions out of the currently regarded set $\mathcal{F}_{i_l}$ of algorithm 3 does not have any impact on the cone robustness degree of still non-dominated solutions and hence prove that algorithm 3 is well-defined, we first need to prove the following lemma.

Lemma D.1
Let $\prec$ be a strict partial order on $\mathbb{R}^k$. Let $F \neq \emptyset$ be a finite non-empty subset of $\mathbb{R}^k$ and for $i \in \mathbb{N}$ let $S_i \subset F$ be a subset with $i = |S_i| < |F|$. Assume that the following assumption holds true.

$$\forall y \in S_i \quad \exists \bar{y} \in F \quad with \quad \bar{y} \prec y. \tag{31}$$

Then it follows

(i) There exists a point $y \in S_i$ and a point $\bar{y} \in F \setminus S_i$ s.t. $\bar{y} \prec y$.

(ii) For all points $y \in S_i$ exists a point $y \in F \setminus S_i$ with $\bar{y} \prec y$.

Proof: We will prove (i) by induction over all $i \in \mathbb{N}$.
Start: Case $i = 1$ is clear due to the irreflexivity of $\prec$. Case $i = 2$ is clear as well, due to the transitivity and irreflexivity of $\prec$.
Step: $i \rightsquigarrow i + 1$: Assume the statement to hold for all $i_0 \leq i$ for some $i \in \mathbb{N}$. We prove it for $i + 1$ as well. Let y_0 be in S_{i+1}. Due to (31) there either exists a $\hat{y} \in F \setminus S_{i+1}$ with $\hat{y} \prec y_0$, then we are done. Assume such an $\hat{y}$ does not exist, then there has to exist a $y_1 \in S_{i+1}$ with

$$y_1 \prec y_0, \tag{32}$$

due to (31). We define $\bar{S}_i := S_{i+1} \setminus \{y_0\}$ and assume as well, that

$$\nexists y^* \in \bar{S}_i, y \in F \setminus S_{i+1} : \quad y \prec y^*, \tag{33}$$

otherwise we are done with the proof as well. Furthermore by induction exists a $\tilde{y} \in \bar{S}_i$ and a $\bar{y} \in F \setminus \bar{S}_i$ s.t.

$$\bar{y} \prec \tilde{y}. \tag{34}$$

If we show that $\bar{y} \neq y_0$, we are done as then $\bar{y} \in F \setminus S_{i+1}$ would hold true, which is a contradiction to (33). Assume the opposite, i.e.

$$\bar{y} = y_0, \tag{35}$$

and prove by contradiction, as the transitivity of $\prec$ will be violated.
For y_1 we have $y_1 \in S_i^{(1)} := \bar{S}_i \setminus \big(\{\tilde{y}\} \cup \{y \in \bar{S}_i | y_0 \prec y\}\big)$, otherwise we violate the transitivity doe to the irreflexivity already. Due to (31) and (33) there exists a $y_2 \prec y_1, y_2 \in S_i^{(2)} := \bar{S}_i \setminus \big(\{\tilde{y}\} \cup \{y \in \bar{S}_i | y_s \prec y, s \in \{0,1\}\}\big)$ otherwise the transitivity would be violated again. Following this argument inductively, we get a sequence $(y_j)_{j\in\{1,2,\dots,l\}}, l \leq i$ with $y_l \prec y_{l-1} \prec \cdots \prec y_2 \prec y_1$ being element of $S_i^{(j)} := \bar{S}_i \setminus \big(\{\tilde{y}\} \cup \{y \in \bar{S}_i | y_s \prec y, s \in \{1,2,\dots,j-1\}\}\big)$ for all $j \in \{1,2,\dots,l\}$ respectively. As $|\bar{S}_i| < \infty$ and the induction hypothesis holds for all $i_0 \leq i$ all $y \in \bar{S}_i \setminus \{\tilde{y}\}$ are element of some set $\bar{S}_i^{(t)} := \Big\{y \in \bar{S}_i \setminus \{\tilde{y}\} | y_t \prec y, y_t \in S_i^{(t)}\Big\}$ for a $t < l$. As y_l remains and there has to be some $y \in \bar{S}_i$ with $y \prec y_l$ due to (31) and y being an element of $\bar{S}_i^{(t)}$ would violate the transitivity of $\prec$, we have $\tilde{y} \prec y_l$ and with (32), (34) and (35) we get $\tilde{y} \prec y_l \prec y_{l-1} \prec \cdots \prec y_1 \prec y_0 \prec \tilde{y}$, which violates the transitivity doe to the irreflexivity of $\prec$. Hence $\bar{y} \neq y_0$ must hold true, which proves the assertion.

We prove assertion (ii) by induction as well.
Start: For the case $i = 1$ the assertion is clear doe to (31). For $i = 2$ as well due to the transitivity and irreflexivity of $\prec$.
Step: $i \rightsquigarrow i+1$: Assume (ii) holds for all $i_0 \leq i$. Let $y_0 \in S_{i+1}$ hold true. Then there exists a $\bar{y} \in F$ s.t. $\bar{y} \prec y_0$, due to (31). Assume $\bar{y} \notin F \setminus S_{i+1}$, that is equivalent to $\bar{y} \in \bar{S}_i := S_{i+1} \setminus \{y_0\}$ due to the irreflexivity of $\prec$. Then as the existence of a point $\tilde{y} \in F \setminus \bar{S}_i$ with $\tilde{y} \prec \bar{y}$ follows from induction, it follows by the transitivity of $\prec$ that $\tilde{y} \prec y_0$ has to hold true. This proves the assertion ■

A direct consequence of lemma D.1 is the following corollary.

Corollary D.2
Let $\mathcal{F}_1$, $\mathcal{F}_{i_l} \subseteq \mathcal{F}_1$, $(\delta_{i_l})_{l\in\mathbb{N}}$ and $\bar{L} = \mathcal{F}_1 \setminus \mathcal{F}_{i_l} = \bigcup_{j<l} L_j$ be defined as through algorithm 3 for all $l \in \mathbb{N}$. Then it follows that

$$\bar{L}_l \subseteq \mathcal{F}_{i_l} + \mathrm{c}(K, \delta_{i_l}) \setminus \{0\}$$

holds true for all $l \in \mathbb{N}$.

Proof: Direct consequence of lemma D.1. ■

Now we can prove the actual assertion.

Lemma D.3
Let $\mathcal{F}_1$, $\mathcal{F}_{i_l} \subseteq \mathcal{F}_1$, $(\delta_{i_l})_{l\in\mathbb{N}}$ and $\bar{L} = \mathcal{F}_1 \setminus \mathcal{F}_{i_l} = \bigcup_{j<l} L_j$ be defined as through algorithm 3 for all $l \in \mathbb{N}$. Let $r \in \mathbb{N}, r > l$ and $x_0 \in \mathcal{F}_{i_l}$ with $CRD^{\mathrm{c}}_{\mathcal{F}_{i_l}}(x_0) = \delta_{i_r}$ hold true. Then follows that $CRD^{\mathrm{c}}_{\mathcal{F}_{i_l}}(x_0) = CRD^{\mathrm{c}}_{\mathcal{F}_1}(x_0)$ for all $l \in \mathbb{N}$.

Proof: Suppose there exists an $l \in \mathbb{N}$, a $\delta \in (\delta_{i_l})_{l\in\mathbb{N}}$ with $\delta < \delta_r$ a solution $x \in \bar{L}_l, x \in L_j$ with $j < l, j \in \mathbb{N}$ and $CRD^{\mathrm{c}}_{\mathcal{F}_1}(x_0) = CRD^{\mathrm{c}}_{\mathcal{F}_{i_l}\cup\{x\}}(x_0) = \delta$, which would disprove the assertion. Then it follows with definition 2.5 of the cone robustness degree and corollary 1.18 that

$$f(x_0) \in f(x) + \mathrm{c}(K, \delta) \setminus \{0\} \tag{36}$$

holds true.

Case 1: $\delta \leq \delta_{i_l}$:
Resulting from corollary D.2 there exists a solution $\bar{x} \in \mathcal{F}_{i_l}$ s.t. $f(\bar{x}) \prec_{\mathrm{c}(K,\delta_{i_l})} f(x)$. As $f(\bar{x}) = f(x_0)$ is impossible due to the transitivity and irreflexivity of the binary relation $\prec_{\mathrm{c}(K,\delta_{i_l})}$ on $\mathbb{R}^k$, we have with corollary 1.18, that $f(x) \in f(\bar{x}) + \mathrm{c}(K, \delta_{i_l}) \setminus \{0\}$ for $f(\bar{x}) \neq f(x)$. This means that $f(x) + \mathrm{c}(K, \delta) \setminus \{0\} \subseteq f(\bar{x}) + \mathrm{c}(K, \delta_{i_l}) \setminus \{0\}$ due to definition 2.2 of an ascending cone mapping, as $\mathrm{c}(K, \delta) \subseteq \mathrm{c}(K, \delta_{i_l})$ holds true as well. Together with (36) we gain that $f(x_0) \in f(\bar{x}) + \mathrm{c}(K, \delta_{i_l}) \setminus \{0\}$ which is a contradiction to x_0 being an element of $\mathcal{F}_{i_l}$.

Case 2: $\delta_{i_r} > \delta > \delta_{i_l}$:
Due to $CRD_{\mathcal{F}_{i_l}}(x_0) = \delta_{i_r} > \delta$, the following has to hold true as well,

$$\nexists \tilde{x} \in \mathcal{F}_{i_l} : \quad f(x_0) \in f(\tilde{x}) + \mathrm{c}(K, \delta) \setminus \{0\}. \tag{37}$$

Resulting from corollary D.2 we know that there exists a $\bar{x} \in \mathcal{F}_{i_l}$ s.t. $f(x) \in f(\bar{x}) + \mathrm{c}(K, \delta_{i_l}) \setminus \{0\}$. Hence $f(x) + \mathrm{c}(K, \delta_{i_l}) \setminus \{0\} \subseteq f(\bar{x}) + \mathrm{c}(K, \delta_{i_l}) \setminus \{0\}$ has to hold true and with definition 2.2 of an ascending cone mapping and $\delta > \delta_{i_l}$ we gain that $f(x) + \mathrm{c}(K, \delta) \setminus \{0\} \subseteq f(\bar{x}) + \mathrm{c}(K, \delta) \setminus \{0\}$ has to hold true as well. Together with (36) it follows that $f(x_0) \in f(\bar{x}) + \mathrm{c}(K, \delta)$ which is a contradiction to (37).

From this results that $CRD^{\mathrm{c}}_{\mathcal{F}_{i_l}}(x_0) = CRD^{\mathrm{c}}_{\mathcal{F}_{i_l}\cup\{x\}}(x_0) = CRD^{\mathrm{c}}_{\mathcal{F}_1}(x_0)$, which proves our assertion. ■

Together with the explanations presented in section 2.1.1 after algorithm 3 follows that the algorithm is well-defined.

E List of Java Classes and Interfaces added to jMetal

In this appendix we will present a list of the Java classes and interfaces generated over the course of this dimploma thesis in order to implement the SV-NSGA-II and the CREA. Both algorithms are inserted into the jMetal framework [10], hence all of the implemented modules are applicable for other methaheuristics already existent within jMetal or to develop for the jMetal framework respectively.

E.1 Core Components

The following classes were added to the jmetal.core package within the jMetal framework.

1. SetAlgorithm: Abstract class, from which each set-valued methaheuristic has to inherit from.
2. SetProblem: Abstract class, from which each parametrized multiobjective set-valued optimization problem has to inherit from. More detailed description in section 3.3.2.
3. SolutionSetSet: Class which represents a set of solutions of a set-valued optimization problem. Contains a List of SolutionSet objects each representing one solution of our set-valued problem regarded. More detailed description in section 3.3.2.

E.2 Methaheuristics

The following classes were added to the jmetal.methaheurstics.crea package within the jMetal framework.

1. CREA: The actual methaheuristic described in algorithm 4.
2. CREA_main: The main class of the CREA algorithm. Here the user has to adjust the input parameter setting.

The following classes were added to the jmetal.methaheurstics.nsgaII package within the jMetal framework.

1. SV_NSGAII: The actual methaheuristic described in algorithm 5.
2. SV_NSGAII_main: The main class of the SV-NSGA-II algorithm. Here the user has to adjust the input parameter setting.

E.3 Operators

1. `SetSBXCrossover:` Class which allows the user to perform a simulated binary crossover (SBX) to two solutions of a multiobjective set-valued optimization problem. Gets two `SolutionSet` objects as input parameters of its `execute()` method and inherits form the `Crossover` class. Added to the `jmetal.operators.crossover` package.

2. `SetPolynomialMutation:` Class which allows the user to perform a polynomial mutation to a solution of a multiobjective set-valued optimization problem. Gets a `SolutionSet` object as input parameter of its `execute()` method and inherits form the `Selection` class. Added to the `jmetal.operators.selection` package.

3. `SetBinaryTournament2:` Class which allows the user to perform a binary tournament selection on solutions of a multiobjective set-valued optimization problem. Needs `Comparator` object submitted as input parameter of its constructor, which is able to run on `SolutionSet` objects. Added to the `jmetal.operators.selection` package.

E.4 Problems

The following classes were added to the `jmetal.problems` package within the `jMetal` framework.

1. `Jahn2:` Implementation of the parametrized set-valued optimization problem presented in example 11.

2. `TestProblem1:` Implementation of the multiobjective optimization problem presented in example 2.

E.5 Utility Components

The following classes were added to the `jmetal.util` package within the `jMetal` framework.

1. `SetDistance:` Class containing the `representativeCrowdingDistanceAssignment()` method, whose framework is described in algorithm 6.

2. `SetRanking:` Implementation of an adaptation of the fast non-dominated sorting algorithm introduced in [2] in order to run on sets $\mathcal{F} \subset \mathbb{R}^n$ of solutions of a parametrized set-valued optimization problem $\bar{\mathcal{SP}}^S_{\prec^*_K}$ w.r.t.

a given strict partial order $\prec_K^*$, $* \in \{u, l, s, c, m, mc, mn\}$. Constructor needs a `SolutionSetSet` object representing $\mathcal{F}$ and a `Comparator` object representing $\prec_K^*$ as input parameter.

The following classes were added to the `jmetal.util.comparators` package within the `jMetal` framework.

1. `DegreeDominanceComparator:` Class inheriting from `Comparator` and representing the strict partial order $\prec_{\hat{K}(\delta)}$ on $\mathbb{R}^n$. Where the inducing cone $\hat{K}(\delta) = K(A(\frac{180}{\pi}\delta))$ equals the polyhedral cone of example 1 with $\delta \in \hat{D} = \left[0, \frac{180}{\pi} \arctan\left(\frac{1}{\sqrt{k-1}}\right)\right)$ in arc degrees instead of radians. Its constructor gets a `double` value as input parameter which represents $\delta \in \hat{D}$. Contains the `compare()` method which needs two `Solution` objects as input parameters. These two `Solution` objects represent an ordered pair of solutions (x^1, x^2) of a multiobjective optimization problem $\mathcal{P}_K$. The `compare()` method tests wether x^1 dominates x^2, x^2 dominates x^1 or both are incomparable to each other w.r.t. $\hat{K}(\delta)$. I.e. it returns the integer value -1 iff

$$\begin{aligned} &\forall i \in \{1, 2, \ldots, k\} : && \langle f(x^1), \hat{a}_i(\delta)\rangle \leq \langle f(x_2), \hat{a}_i(\delta)\rangle, \\ &\exists i \in \{1, 2, \ldots, k\} : && \langle f(x^1), \hat{a}_i(\delta)\rangle < \langle f(x_2), \hat{a}_i(\delta)\rangle, \end{aligned}$$

holds true, where $\hat{a}_i(\delta)$ equals the i-th row vector $a_i(\frac{180}{\pi}\delta)$ of the matrix $A(\frac{180}{\pi}\delta)$ of example 1 with to arc degrees transformed domain. The method returns 1 iff

$$\begin{aligned} &\forall i \in \{1, 2, \ldots, k\} : && \langle f(x^1), \hat{a}_i(\delta)\rangle \geq \langle f(x_2), \hat{a}_i(\delta)\rangle, \\ &\exists i \in \{1, 2, \ldots, k\} : && \langle f(x^1), \hat{a}_i(\delta)\rangle > \langle f(x_2), \hat{a}_i(\delta)\rangle, \end{aligned}$$

holds true and 0 otherwise. That this equals a test wether one solution dominates the other one is proven in lemma E.1.

2. `DegreeDominanceComparator2:` Extension of the `DegreeDominanceComparator` class containing the additional method `compareOne()` which as well needs two `Solution` objects representing an ordered pair of solutions (x^1, x^2) as input parameter. The `compareOne()` method tests wether $f(x^1) \preceq_{\hat{K}(A(\delta))} f(x^2)$ holds true or not, where $\preceq_{\hat{K}(A(\delta))}$ is the partial order induced by $\hat{K}(\delta)$ for all $\delta \in \hat{D}$.

The following classes were added to the `jmetal.util.comparators` package within the `jMetal` framework as well.

1. `CertainlyLessOrderComparator`: Class representing the pre-order $\preccurlyeq_K^c$ and the strict partial order $\prec_K^c$.

2. `LowerSetLessOrderComparator`: Class representing the pre-order $\preccurlyeq_K^l$ and the strict partial order $\prec_K^l$.

3. `MinmaxCertainlyLessOrderComparator`: Class representing the pre-order $\preccurlyeq_K^{mc}$ and the strict partial order $\prec_K^{mc}$.

4. `MinmaxCertainlyNondominatedOrderComparator`: Class representing the pre-order $\preccurlyeq_K^{mn}$ and the strict partial order $\prec_K^{mn}$.

5. `MinmaxLessOrderComparator`: Class representing the pre-order $\preccurlyeq_K^l$ and the strict partial order $\prec_K^l$.

6. `SetLessOrderComparator`: Class representing the pre-order $\preccurlyeq_K^s$ and the strict partial order $\prec_K^s$.

7. `UpperSetLessOrderComparator`: Class representing the pre-order $\preccurlyeq_K^u$ and the strict partial order $\prec_K^u$.

Each of theses seven classes inherits from `Comparator` and contains the methods `compareOne()`, `compareStrict()` and `compare()` which all need two `SolutionSet` objects as input parameters. These two `SolutionSet` objects represent an ordered pair (x^1, x^2) of solutions of a parametrized set-valued optimization problem $\mathcal{SP}_K^S$. The method `compareOne()` tests if $F(x^1) \preccurlyeq_K^* F(x^2)$ holds true or not. The method `compareStrict()` tests if $F(x^1) \prec_K^* F(x^2)$ holds true or not and hence tests wether x^1 dominates x^2. And the method `compare()` tests if x^1 dominates x^2, x^2 dominates x^1 or if both are incomparable to each other w.r.t. $\prec_K^*$ for all $* \in \{u, l, s, c, m, mn, mc\}$. Here the cone K equals $\hat{K}(\delta)$ as the constructor needs a `DegreeDominanceComparator2` object as input parameter.
The following classes were added to the `jmetal.util.comparators` package within the `jMetal` framework as well.

1. `OverallConstraintViolationSetComparator`: Performs a constraint violation comparison and checks for violated constraints as explained in 2.1.1 after algorithm 4. Contains the `compare()` method expecting two `SolutionSet` objects as input parameters and performs the actual constraint violation comparison and contains the `needToCompare()` method, expecting one `SolutionSet` object and checks for violated constraints of the solution represented by this object. Implements the `IConstraintViolationSetComparator` interface.

2. `SetCrowdingComparator:` Implements the `Comparator` interface in order to compare two `SolutionSet` objects based on their assigned crowding distance values.

3. `SetRankComparator:` Implements the `Comparator` interface in order to compare two `SolutionSet` objects based on their rank. Necessary for the `SetCrowdingComparator` class.

Lemma E.1
Let $y_1, y_2 \in \mathbb{R}^k$ *and* $\delta \in D = [0, \arctan(\frac{1}{\sqrt{k-1}}))$ *hold true. Let* $K(A(\delta))$ *be the* k*-edged cone induced by the matrix* $A(\delta) \in \mathbb{R}^{k \times k}$ *defined as in example 1 and let* $a_i(\delta)$ *be the* i*-th row vector of* $K(A(\delta))$*. Then does* $y_1 \prec_{K(A(\delta))} y_2$ *hold true iff*

$$\begin{aligned} \forall i \in \{1, 2, \ldots, k\} : \quad & \langle y_1, a_i(\delta) \rangle \leq \langle y_2, a_i(\delta) \rangle, \\ \exists i \in \{1, 2, \ldots, k\} : \quad & \langle y_1, a_i(\delta) \rangle < \langle y_2, a_i(\delta) \rangle, \end{aligned}$$

holds true.

Proof: Let $y_1 \prec_{K(A(\delta))} y_2$ hold true. Since $K(A(\delta))$ is pointed and convex it follows with corollary 1.6 point 2.(ii) that this is equal to $y_1 \preceq_{K(A(\delta))} y_2$ and $y_1 \neq y_2$ holding true. With definition 1.9 of a polyhedral cone and definition 1.5 of our ordering relation, this again is equivalent to

$$\begin{aligned} & y_2 - y_1 \in \bigcap_{i=1}^{k} H_i(A(\delta)) \quad \text{and} \quad y_1 \neq y_2 && \iff \\ & \forall i \in \{1, 2, \ldots, k\} : \ \langle y_2 - y_1, a_i(\delta) \rangle \geq 0 \quad \text{and} \quad y_1 \neq y_2 && \iff \\ & \forall i \in \{1, 2, \ldots, k\} : \ \langle y_2, a_i(\delta) \rangle - \langle y_1, a_i(\delta) \rangle \geq 0 \quad \text{and} \quad y_1 \neq y_2 && \iff \\ & \forall i \in \{1, 2, \ldots, k\} : \ \langle y_2, a_i(\delta) \rangle \geq \langle y_1, a_i(\delta) \rangle \quad \text{and} \quad y_1 \neq y_2. \end{aligned}$$

This equals the assertion. ■