Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Mathematical Sciences

Axel Henæs Rønold

Methods in Uncertain Multi-Objective Optimization

Master's thesis in Industrial Mathematics
Supervisor: Elisabeth Anna Sophia Köbis
July 2021

NTNU
Norwegian University of
Science and Technology

Axel Henæs Rønold

Methods in Uncertain Multi-Objective Optimization

Master's thesis in Industrial Mathematics
Supervisor: Elisabeth Anna Sophia Köbis
July 2021

Norwegian University of Science and Technology
Faculty of Information Technology and Electrical Engineering
Department of Mathematical Sciences

NTNU
Norwegian University of
Science and Technology

Abstract

We begin the paper by discussing concepts of efficiency for uncertain multi-objective optimization problems by using different set order relations. In all, we discuss four different relations: the upper set less order relation introduced by Kuroiwa [1], the lower set less order relation introduced by Kuroiwa [1], the set less order relation introduced by Young [2] and the strict set less order relation which was introduced as alternative set less order relation by Ide et al. [3]. We then discuss the different characteristics of these order relations and the differences between them. After this, we look at a method that has been used to solve multi-objective optimization problems where the feasible set is non-convex and even disconnected, the weighted constraint method introduced by Burachik et al. [4]. The next part shows that we can use the weighted constraint method to solve uncertain multi-objective optimization problems. Hence, it is possible to obtain upper set less order efficient solutions of a problem by solving the weighted constraint method. Lastly, we wanted to explore how the weighted constraint method performed when comparing it to other methods that have been previously used on uncertain multi-objective optimization problems. In our comparison, we used the weighted sum method. The investigation focuses on problems with two and three objectives. It shows that the weighted constraint method is quicker to compute than the weighted sum method and performs better when the number of feasible elements was low. However, when the number of feasible elements increased, the weighted sum method found more upper set less order efficient solutions.

Contents

List of Figures

Chapter 1

Introduction

The main topic we are focusing on in this paper is uncertain multi-objective optimization. Uncertainty is common when we are dealing with optimization problems in the real world. When we decide between the different options in a problem, we are not always sure how the future will unfold, but it still influences our choice. When we meet uncertainty, the literature has suggested two main approaches. One is stochastic optimization, where the uncertain parameters are assumed to possess a probability distribution. We then optimize the expected value of each option while they have other outcomes which are still possible to occur with some probability. The other approach is the one we are going to focus on, namely robust optimization. In this approach, we consider the case where we have no stochastic information about the uncertain parameters. There are different variations of what a robust solution is. One of the concepts that have been introduced is minmax robustness, which was firstly introduced by Soyster [5]. With this approach, the goal is to find solutions that are feasible for every future scenario. Hence, the objective becomes to optimize the worst-case scenario for each alternative. What we want to accomplish by finding robust solutions is to have solutions that are less sensitive to perturbations in the data. Let us explain with a scenario. Imagine we are a company presented with two different plans. The first one earns the company much money if everything goes right but bankrupts the company if something suddenly goes wrong. The other one is not that lucrative, but a very safe solution where we know that very little can go wrong. The first solution is very sensitive and hence not very robust as it becomes non-feasible for some scenarios. However, if we optimize by just looking at the output and not factoring in the risk, we would have gone for this option.

These types of choices are the ones we try to eliminate by optimizing on the worst-case scenario as we do in robust optimization. When we do this type of optimization, it becomes a set-based method because we assume that the uncertain parameters belong to an uncertainty set known before solving the optimization problem.

We will discuss this further in the paper, namely to use different set order relations to define the robust solutions of uncertain multi-objective optimization problems. In the example when we optimize on the worst-case scenario, we used a set order relation named in the literature as the upper set less order relation [1]. This relation has a pessimistic approach as we hedge against the scenarios with the worst outcomes. In addition to this one, we will discuss other set order relations to define different approaches a decision-maker can use, for instance, a more optimistic approach. Throughout the discussions on these set order relations, we are going to look at uncertain multi-objective problems where the objective functions are affected by uncertain data, which is given by an arbitrary uncertainty set $\mathcal{U}$. All possible scenarios of the uncertain input data are represented within this set. We want to show that we can end up with varying sets of efficient solutions for the same uncertain multi-objective optimization problem by using the different definitions of set order relations. For each set order relation, we will discuss how it affects the set of efficient solutions. We will also show two methods that can be used to find the different efficient solutions, the weighted sum method, and the ϵ-constraint method.

However, as we will show in the paper, it is easy to construct uncertain multi-objective optimization problems where methods such as the two aforementioned can not obtain all the different efficient solutions given the different set order relations. In an attempt to solve these issues, we are going to look at another method that has been used in multi-objective optimization, namely the weighted constraint method [4]. By first showing that solving the problem with this method produces the solutions we are looking for, we want to investigate how it performs versus other methods used in solving uncertain multi-objective optimization problems to see if it can find more solutions.

1.1 Clarification of reuse of text

This work is a continuation of the work done in my project thesis. In my project thesis I gathered theory on uncertain multi-objective optimization and also on the weighted constraint method in order to do the work I have done now in my master thesis, solving uncertain multi-objective problems with the weighted constraint method. The theory is the base of all this work and is central in communicating to you, the reader, what the work I now have done on the master thesis is. I put in a lot of effort to write the project thesis and since it is the foundation of the work in my master thesis I feel it is appropriate to use the text here. I have therefore reused the text from that thesis in this paper. I have made some improvements of the text and also made changes as some of the work turned out not to be of focus in my master thesis. However, most of these chapters are still very similar to how they were presented in my project thesis because they are relevant in their current shape. This regards to Chapter 1 through 4. [6]

Chapter 2

Preliminaries

Firstly, we need to introduce some notation on multi-objective optimization. Given a feasible set $\mathcal{X} \subseteq \mathbb{R}^n$ defined by some constraints, we want to minimize a function $f : \mathcal{X} \to \mathbb{R}^k$. We can write it more formally as

$$\begin{aligned} \textbf{min} \quad & f(x) \\ \textbf{s.t.} \quad & x \in \mathcal{X}. \end{aligned} \tag{$\mathcal{P}$}$$

Since we are comparing solutions in $\mathbb{R}^k$ where $k > 1$, it is necessary to define relations to compare them as it lack total order. To do this, we use the relations $\{\leqq, \leq, <\}$, referred to in Ehrgott [7]. Let $\{y_1, y_2\} \in \mathbb{R}^k$, then we say that

$$\begin{aligned} y_1 \leqq y_2 &\iff y_2^i \in [y_1^i, \infty) \ \forall \ i \in 1, ..., k \\ y_1 \leq y_2 &\iff y_1 \leqq y_2, y_1 \neq y_2 \\ y_1 < y_2 &\iff y_2^i \in (y_1^i, \infty) \ \forall \ i \in 1, ..., k. \end{aligned}$$

Furthermore, we define the ordering cones $\left\{\mathbb{R}^k_{\leqq}, \mathbb{R}^k_{\leq}, \mathbb{R}^k_{>}\right\}$ as

$$\mathbb{R}^k_{[\leqq, \leq, >]} := \left\{ x \in \mathbb{R}^k : x [\leqq, \leq, >] 0 \right\}.$$

With this ordering, we want to find all feasible solutions $x \in \mathcal{X}$ to $(\mathcal{P})$ that are [*strictly*/ $\cdot$ / *weakly*] *efficient*, which means that its function value, $f(x)$, is not

dominated by any other function value, $f(\hat{x})$, from a point $\hat{x} \in \mathcal{X} \setminus \{x\}$. We can write this as

$$x \text{ is [strictly/} \cdot \text{/weakly] efficient} \iff \nexists \hat{x} \in \mathcal{X} \setminus \{x\} : f(\hat{x}) \in f(x) - \mathbb{R}^k_{[\geqq,\geq,>]}.$$

Remark 2.1. A strictly efficient point is also an efficient point. An efficient point is also a weakly efficient point, hence

strictly efficient $\implies$ efficient $\implies$ weakly efficient.

Given the tools already presented, we want to define the uncertain counterpart for multi-objective optimization. Given a set of scenarios $\mathcal{U} \subseteq \mathbb{R}^m$, also referred to as the uncertainty set, an uncertain multi-objective optimization problem is given as the family $(\mathcal{P}(\mathcal{U}), \xi \in \mathcal{U})$ of multi-objective optimization problems

$$\begin{aligned} \textbf{min} \quad & f(x,\xi) \\ \textbf{s.t.} \quad & x \in \mathcal{X}, \end{aligned} \qquad \mathcal{P}(\mathcal{U})$$

with objective function $f : \mathbb{R}^n \times \mathcal{U} \to \mathbb{R}^k$, a feasible set $\mathcal{X} \subseteq \mathbb{R}^n$ and $\xi \in \mathcal{U}$ to represent one particular scenario of the uncertainty set. From this framework, it is clear that we need to extend our definitions to define what an efficient solution is. The reason being that uncertain optimization is, by our definition, a family of problems where the object changes by each scenario $\xi \in \mathcal{U}$.

Hence, we define the set

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

which is the set of all possible objective values for a point $x \in \mathcal{X}$ given each $\xi \in \mathcal{U}$. We now need to find out how to turn the family of uncertain values for each feasible point into a deterministic optimization formulation. One way to do this is to define different set order relations to define what property a feasible set $f_{\mathcal{U}}(x)$ needs to have in order to dominate another feasible set $f_{\mathcal{U}}(\hat{x})$ given all feasible points from $x \in \mathcal{X}$ and $\hat{x} \in \mathcal{X} \setminus \{x\}$.

Chapter 3

Definitions of efficiency based on set order relations

In this section, we want to introduce different set order relations to define efficient solutions in uncertain multi-objective optimization problems based on various approaches. We will also discuss how this affects the properties affiliated with the solutions that the different relations give us. To guide us through the different set order relations in the chapter, we use Example 3.1.

Example 3.1. Figure 3.1 illustrates an uncertain multi-objective optimization problem for $k = 2$, hence with $\mathbb{R}^2_{\geq}$ as ordering cone.

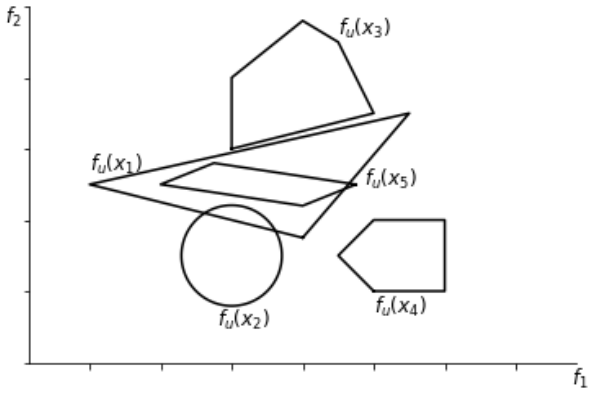


Figure 3.1: Uncertain multi-objective optimization problem with feasible set $\mathcal{X} = \{x_1, x_2, x_3, x_4, x_5\}$.

3.1 Upper set less order relation

3.1.1 Description and problem formulation

The first set order relation we want to define, introduced by Kuroiwa [1], is the upper set less order relation:

Definition 3.1. A set $A \subseteq \mathbb{R}^k$ *dominates* a set $B \subseteq \mathbb{R}^k$ with respect to the upper set less order relation, denoted $A \preceq^u_{[\leqq,\leq,<]} B$, with respect to $\mathbb{R}^k_{[\leqq,\leq,<]}$ if

$$A \preceq^u_{[\leqq,\leq,<]} B \iff A \subseteq B - \mathbb{R}^k_{[\leqq,\leq,<]}.$$

Remark 3.1. The relation can be equivalently written as

$$A \preceq^u_{[\leqq,\leq,<]} B \iff \forall a \in A \exists b \in B : a [\leqq, \leq, <] b.$$

The way to frame $\mathcal{P}(\mathcal{U})$ to mirror this property is to take the supremum of the uncertainty set

$$\min_{x \in \mathcal{X}} \sup_{\xi \in \mathcal{U}} \quad f(x, \xi). \tag{$\mathcal{P}(\xi)^{\mathrm{u}}$}$$

3.1.2 Efficiency and interpretation

Definition 3.2. Given an uncertain multi-objective optimization problem $\mathcal{P}(\mathcal{U})$, a solution $x \in \mathcal{X}$ to $\mathcal{P}(\mathcal{U})$ is *upper set less ordered* [*strictly*/ $\cdot$ / *weakly*] *efficient* if there is no $\bar{x} \in \mathcal{X} \setminus \{x\}$ s.t. $f_{\mathcal{U}}(\bar{x}) \preceq^u_{[\leqq,\leq,<]} f_{\mathcal{U}}(x)$ with respect to $\mathbb{R}^k_{[\leqq,\leq,<]}$, or equivalently written

$$\nexists \bar{x} \in \mathcal{X} \setminus \{x\} : f_{\mathcal{U}}(\bar{x}) \subseteq f_{\mathcal{U}}(x) - \mathbb{R}^k_{[\leqq,\leq,<]}.$$

The way to interpret this set order relation is that a solution set $f_{\mathcal{U}}(x)$ is efficient if there does not exist another solution set $f_{\mathcal{U}}(\bar{x})$ such that the worst case scenario for $\bar{x}$ is better than the worst case scenario for x for the given problem. We can look at this set ordering as pessimistic as the efficient solutions to the problem given the feasible sets $f_{\mathcal{U}}(x), x \in \mathcal{X}$ will be chosen on the grounds of which ones have the greatest worst case scenario. Hence, this approach can be used to find risk averse solutions. In our example, we

can see that only $f_{\mathcal{U}}(x_2)$ and $f_{\mathcal{U}}(x_4)$ fulfill the criteria. Looking at Figure 3.3 we observe that $f_{\mathcal{U}}(x_1) - \mathbb{R}^2_{[\leqq,\geq,>]}$, $f_{\mathcal{U}}(x_3) - \mathbb{R}^2_{[\leqq,\geq,>]}$ and $f_{\mathcal{U}}(x_5) - \mathbb{R}^2_{[\leqq,\geq,>]}$ contains $f_{\mathcal{U}}(x_2)$. Hence, only x_2 and x_4 are upper set less ordered strictly efficient.

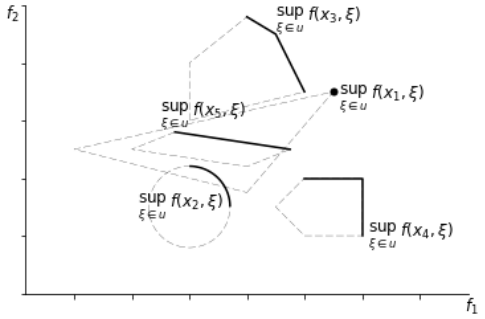


Figure 3.2: Supremum of every feasible set.

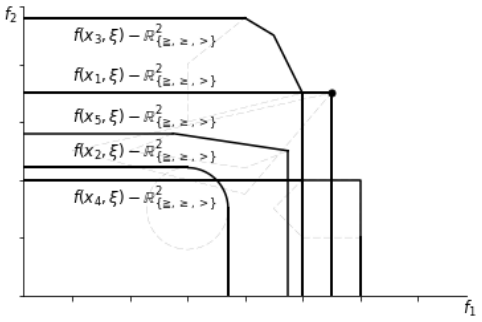


Figure 3.3: All five sets when we subtract $\mathbb{R}^2_{[\leqq,\geq,>]}$.

3.1.3 Computing upper set less ordered efficient solutions

We can use approaches from deterministic multi-objective optimization to compute upper set less ordered efficient solutions by extending their framework from deterministic to uncertain.

Weighted sum scalarization

This method forms a single objective optimization problem by multiplying each objective function by some non-negative weight λ_i and summing them together. So with a weight vector $\lambda \in \mathbb{R}^k_{[\geq,>]}$, we consider

$$\min_{x\in\mathcal{X}} \quad \sum_{i=1}^{k} \lambda_i f_i(x) \qquad \mathcal{P}_\lambda$$

The way we extend the framework to use this method to compute upper set less ordered efficient solutions is to insert the problem formulation obtained in 3.1.1, $\mathcal{P}(\xi)^{\mathrm{u}}$:

$$\min_{x\in\mathcal{X}} \sup_{\xi\in\mathcal{U}} \quad \sum_{i=1}^{k} \lambda_i f_i(x,\xi) \qquad \mathcal{P}(\mathcal{U})^{\mathrm{u}}_{\lambda}$$

Theorem 3.1. ***(Theorem 4.3. Ehrgott et al. [8])*** *Given an uncertain multi-objective optimization problem $\mathcal{P}(\mathcal{U})$, the following statements hold.*

1. *If $\hat{x} \in \mathcal{X}$ is the unique optimal solution to $\mathcal{P}(\mathcal{U})^u_\lambda$ for some $\lambda \in \mathbb{R}^k_{\geq}$, then $\hat{x}$ is upper set less ordered strictly efficient solution to $\mathcal{P}(\mathcal{U})$.*
2. *If $\hat{x} \in \mathcal{X}$ is an optimal solution to $\mathcal{P}(\mathcal{U})^u_\lambda$ for some $\lambda \in \mathbb{R}^k_{\{\geq,>\}}$ and*

$$\max_{\xi \in \mathcal{U}} \sum_{i=1}^{k} \lambda_i f_i(x, \xi)$$

exists for all $x \in \mathcal{X}$, then $\hat{x}$ is upper set less ordered [·/weakly] efficient solution to $\mathcal{P}(\mathcal{U})$.

Proof. 1. Assume $\hat{x}$ is not upper set less ordered [strictly/·/weakly] efficient for $\mathcal{P}(\mathcal{U})$. Then there exists an $x' \in \mathcal{X}$ such that

$$f_{\mathcal{U}}(x') \subseteq f_{\mathcal{U}}(\hat{x}) - \mathbb{R}^k_{[\leqq,\leq,<]} \implies \forall \xi \in \mathcal{U} \exists \eta \in \mathcal{U} : f(x', \xi)[\leqq, \leq, <] f(\hat{x}, \eta)$$

Now choose $\lambda \in \mathbb{R}^k_{[\leqq,\leq,<]}$ arbitrary but fixed.

$$\begin{aligned}
&\implies \forall \xi \in \mathcal{U} \exists \eta \in \mathcal{U} : \sum_{i=1}^{k} \lambda_i f(x', \xi)[\leqq, \leq, <] \sum_{i=1}^{k} \lambda_i f(\hat{x}, \eta) \\
&\iff \forall \xi \in \mathcal{U} : \sum_{i=1}^{k} \lambda_i f(x', \xi)[\leqq, \leq, <] \sup_{\eta' \in \mathcal{U}} \sum_{i=1}^{k} \lambda_i f(\hat{x}, \eta') \\
&\iff \sup_{\xi' \in \mathcal{U}} \sum_{i=1}^{k} \lambda_i f(x', \xi')[\leqq, \leq, <] \sup_{\eta' \in \mathcal{U}} \sum_{i=1}^{k} \lambda_i f(\hat{x}, \eta')
\end{aligned}$$

The last equivalence holds because for 2.

$$\max_{\xi' \in \mathcal{U}} \sum_{i=1}^{k} \lambda_i f(x', \xi')$$

exists. However, this means that $\hat{x}$ is not [the unique/an/an] optimal solution to $\mathcal{P}(\mathcal{U})^{\mathrm{u}}_\lambda$ for $\lambda \in \mathbb{R}^k_{[\leqq,\leq,<]}$. □

Given a set of scalarization vectors Λ, we can now compute upper set less ordered efficient solutions by solving $\mathcal{P}(\mathcal{U})^{\mathrm{u}}_\lambda$ for every $\lambda \in \Lambda$. One challenge with the method is the choice of Λ. On the other hand, the technique does not add any additional constraints to the problem formulation and thus preserves the problem structure.

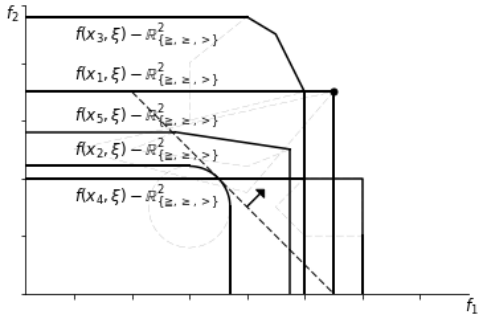


Figure 3.4: Examples of weights to find x_2 as upper set less strictly efficient. Here $\lambda = [1/2, 1/2]$.

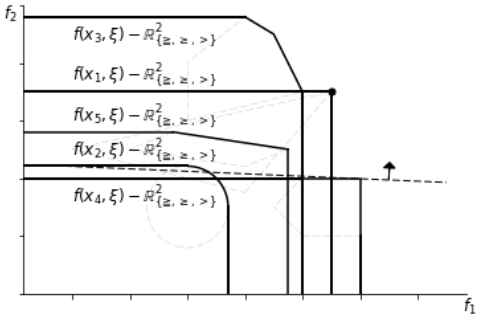


Figure 3.5: Examples of weights to find x_4 as upper set less strictly efficient. Here $\lambda = [3/73, 70/73]$.

ϵ-constraint scalarization

This approach uses the idea of minimizing one of the objective functions, the i^{th} objective function, while the others are less than a value ϵ_j, $j \neq i$. By doing this for every value from $i \in \{1, ..., k\}$ and $\epsilon \in \mathbb{R}^k_{\geq}$, we consider the problem formulation

$$\begin{aligned} \min_{x \in \mathcal{X}} \quad & f_i(x) \\ \textbf{s.t.} \quad & f_i(x) \leq \epsilon_j \forall\, j \neq i. \end{aligned} \qquad \mathcal{P}_{(\epsilon,i)}$$

The way we extend the framework to use this method to compute upper less ordered efficient solutions is to insert the problem formulation obtained in 3.1.1, $\mathcal{P}(\xi)^{\mathrm{u}}$:

$$\begin{aligned} \min_{x \in \mathcal{X}} \sup_{\xi \in \mathcal{U}} \quad & f_i(x, \xi) \\ \textbf{s.t.} \sup_{\xi \in \mathcal{U}} \quad & f_i(x, \xi) \leq \epsilon_j \forall\, i \neq j. \end{aligned} \qquad \mathcal{P}(\mathcal{U})^{\mathrm{u}}_{(\epsilon,i)}$$

Theorem 3.2. ***(Theorem 4.7. Ehrgott et al. [8])*** *Given an uncertain multi-objective optimization problem $\mathcal{P}(\mathcal{U})$, the following statements hold.*

1. *If $\hat{x} \in \mathcal{X}$ is the unique optimal solution to $\mathcal{P}(\mathcal{U})^u_{(\epsilon,i)}$ for some $\epsilon \in \mathbb{R}^k$ and some $i \in \{1, .., k\}$, then $\hat{x}$ is upper set less strictly efficient solution to $\mathcal{P}(\mathcal{U})$.*
2. *If $\hat{x} \in \mathcal{X}$ is an optimal solution to $\mathcal{P}(\mathcal{U})^u_{(\epsilon,i)}$ for some $\epsilon \in \mathbb{R}^k$ and some $i \in \{1, .., k\}$ and*

$$\max_{\xi \in \mathcal{U}} f_i(x, \xi)$$

exists for all $x \in \mathcal{X}$, then $\hat{x}$ is upper set less weakly efficient solution to $\mathcal{P}(\mathcal{U})$.

Proof. 1. Assume $\hat{x}$ is not upper set less ordered strictly efficient for $\mathcal{P}(\mathcal{U})$. Then there exists an $x' \in \mathcal{X}$ such that

$$\begin{aligned} f_{\mathcal{U}}(x') \subseteq f_{\mathcal{U}}(\hat{x}) - \mathbb{R}^k_{\leqq} &\implies \forall \xi \in \mathcal{U} \exists \eta \in \mathcal{U} : f(x', \xi) \leqq f(\hat{x}, \eta) \\ &\implies \sup_{\xi' \in \mathcal{U}} f(x', \xi') \leqq \sup_{\eta' \in \mathcal{U}} f(\hat{x}, \eta') \text{ and} \\ &\quad \forall \xi \in \mathcal{U} \exists \eta \in \mathcal{U} : f_j(x', \xi) \leqq f_j(\hat{x}, \eta) \leqq \epsilon_j, j \neq i. \end{aligned}$$

In this scenario, x' is feasible for $\mathcal{P}(\mathcal{U})^{\mathrm{u}}_{(\epsilon,i)}$ and has an equal or better objective value than $\hat{x}$. This is a contradiction to the assumption that $\hat{x}$ is the unique optimal solution to $\mathcal{P}(\mathcal{U})^{\mathrm{u}}_{(\epsilon,i)}$.

2. Assume $\hat{x}$ is not upper set less ordered weakly efficient for $\mathcal{P}(\mathcal{U})$. Then there exists an $x' \in \mathcal{X}$ such that

$$\begin{aligned} f_{\mathcal{U}}(x') \subseteq f_{\mathcal{U}}(\hat{x}) - \mathbb{R}^k_{>} &\implies \forall \xi \in \mathcal{U} \exists \eta \in \mathcal{U} : f(x', \xi) < f(\hat{x}, \eta) \\ &\implies \max_{\xi' \in \mathcal{U}} f(x', \xi') < \max_{\eta' \in \mathcal{U}} f(\hat{x}, \eta') \text{ and} \\ &\quad \forall \xi \in \mathcal{U} \exists \eta \in \mathcal{U} : f_j(x', \xi) < f_j(\hat{x}, \eta) \leqq \epsilon_j, j \neq i. \end{aligned}$$

In this scenario, x' is feasible for $\mathcal{P}(\mathcal{U})^{\mathrm{u}}_{(\epsilon,i)}$ and has a better objective value than $\hat{x}$. This is a contradiction to the assumption that $\hat{x}$ is an optimal solution to $\mathcal{P}(\mathcal{U})^{\mathrm{u}}_{(\epsilon,i)}$. □

Given a set $\mathcal{E}$ of vectors $\epsilon \in \mathbb{R}^k_{\leqq}$, we can now compute upper set less ordered efficient solutions by solving $\mathcal{P}(\mathcal{U})^{\mathrm{u}}_{(\epsilon,i)}$ for each $i \in \{1, ..., k\}$ and every $\epsilon \in \mathcal{E}$. One challenge with the method is to choose the set $\mathcal{E}$ correctly. If the elements in $\mathcal{E}$ are chosen too small, then the set of feasible solutions may be empty. Still, if the elements in $\mathcal{E}$ are chosen too large, then the optimality of the functions representing the constraints decreases.

Remark 3.2. ϵ-constraint method can find all the efficient solutions to a deterministic multi-objective optimization problem, but this is not necessarily the case for uncertain multi-objective optimization problems - even when the sets are convex. To illustrate this, we can look at Example 3.2.

Another problem with this approach lies in the altered problem structure. We add constraints to the problem; hence the problem structure of the original problem is not preserved. This may further complicate the decision process.

Example 3.2. Figure 3.6 shows an uncertain multi-objective optimization problem for $k = 2$ with both x_1 and x_2 as upper set less strictly efficient solutions.

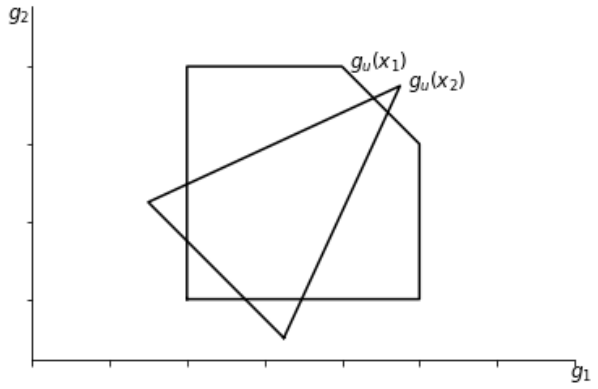


Figure 3.6: Uncertain multi-objective optimization problem with feasible set $\mathcal{X} = \{x_1, x_2\}$.

From what we have learned, we observe in Figure 3.6 that both x_1 and x_2 are upper set less strictly efficient. However, since x_2 has a lower supremum in both objective functions individually and is therefore feasible for whenever x_1 is. As shown in Figure 3.7, we can thus not obtain x_1 as an upper set less strictly efficient solution with this method, even though it is.

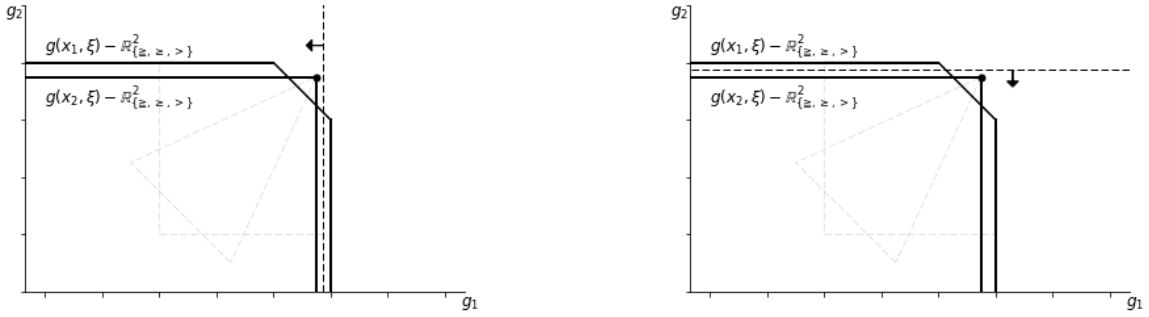


Figure 3.7: Illustration of how x_2 is always feasible for every value $\epsilon \in \mathcal{E}$ where x_1 is feasible for both objective functions. Hence, it is not possible to obtain x_1 as an upper set less strictly efficient solution by using the ϵ-constraint scalarization in this example.

3.2 Lower set less order relation

3.2.1 Description and problem formulation

The second set order relation we are looking at is the lower set less order relation, first introduced by Kuroiwa [1].

Definition 3.3. A set $A \subseteq \mathbb{R}^k$ *dominates* a set $B \subseteq \mathbb{R}^k$ with respect to the lower set less order relation, denoted $A \preceq^l_{[\leqq,\leq,<]} B$, with respect to $\mathbb{R}^k_{[\leqq,\leq,<]}$ if

$$A \preceq^l_{[\leqq,\leq,<]} B \iff A + \mathbb{R}^k_{[\leqq,\leq,<]} \supseteq B.$$

Remark 3.3. The relation can be equivalently be written as

$$A \preceq^l_{[\leqq,\leq,<]} B \iff \forall b \in B \exists a \in A : a [\leqq, \leq, <] b.$$

This time, the way to frame the problem to obtain this property is to take the infimum on the uncertainty set

$$\min_{x \in \mathcal{X}} \inf_{\xi \in \mathcal{U}} \quad f(x, \xi). \tag{$\mathcal{P}(\mathcal{U})^{\text{l}}$}$$

3.2.2 Efficiency and interpretation

Definition 3.4. Given an uncertain multi-objective optimization problem $\mathcal{P}(\mathcal{U})$, a solution $x \in \mathcal{X}$ to $\mathcal{P}$ is *lower set less ordered* [*strictly*/ $\cdot$ /*weakly*] *efficient* if there is no $\bar{x} \in \mathcal{X} \setminus \{x\}$ s.t. $f_{\mathcal{U}}(\bar{x}) \preceq^l_{[\leqq,\leq,<]} f_{\mathcal{U}}(x)$ with respect to $\mathbb{R}^k_{[\leqq,\leq,<]}$, or equivalently written

$$\nexists \bar{x} \in \mathcal{X} \setminus \{x\} : f_{\mathcal{U}}(\bar{x}) + \mathbb{R}^k_{[\leqq,\leq,<]} \supseteq f_{\mathcal{U}}(x).$$

When we examine the solution sets $f_{\mathcal{U}}(x)$ for this set order relation, we observe that in order for a solution to be efficient, there can not exist another solution set $f_{\mathcal{U}}(\bar{x})$ such that the best case scenario for $\bar{x}$ is better than the best case scenario for x. Hence, we can interpret this ordering as optimistic as the efficient solutions to the problem given the feasible sets $f_{\mathcal{U}}(x), x \in \mathcal{X}$ will be evaluated based on greatest best case scenario. Because of this, it is possible to obtain solutions for situations where we are looking

to be risk seeking. For our example, the risk seeking solutions are therefore $f_{\mathcal{U}}(x_1)$ and $f_{\mathcal{U}}(x_2)$. Conversely, as seen in Figure 3.9, $f_{\mathcal{U}}(x_3)$ and $f_{\mathcal{U}}(x_5)$ are contained in $f_{\mathcal{U}}(x_1) + \mathbb{R}^2_{[\geqq,\geq,>]}$ while $f_{\mathcal{U}}(x_3)$ and $f_{\mathcal{U}}(x_4)$ are contained in $f_{\mathcal{U}}(x_2) + \mathbb{R}^2_{[\geqq,\geq,>]}$. Hence, only x_1 and x_2 are lower set less ordered strictly efficient.

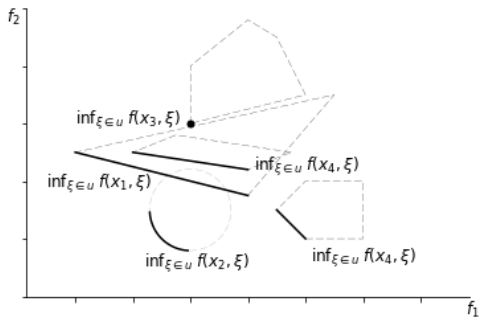


Figure 3.8: Infimum of every feasible set.

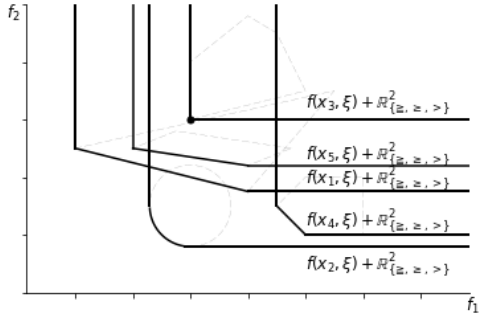


Figure 3.9: All five sets when we add $\mathbb{R}^2_{[\geqq,\geq,>]}$.

3.2.3 Computing lower set less ordered efficient solutions

To compute lower set less ordered efficient solutions, we can use the same extension of a framework as for upper set less ordered efficient solutions in 3.1.3.

Weighted sum scalarization

The way we extend the framework from $\mathcal{P}_\lambda$ to use this method for computing lower set less ordered efficient solutions is to insert the problem formulation obtained in 3.2.1, $\mathcal{P}(\mathcal{U})^{\text{l}}$:

$$\min_{x\in\mathcal{X}} \inf_{\xi\in\mathcal{U}} \quad \sum_{i=1}^{k} \lambda_i f_i(x,\xi) \qquad \mathcal{P}(\mathcal{U})^{\text{l}}_{\lambda}$$

As for the upper set less relation, given a set of scalarization vectors Λ, we can now compute lower set less ordered efficient solutions by solving $\mathcal{P}(\mathcal{U})^{\text{l}}_{\lambda}$ for every $\lambda \in \Lambda$.

Theorem 3.3. ***(Theorem 11 Ide et al. [9])*** *Given an uncertain multi-objective optimization problem $\mathcal{P}(\mathcal{U})$, the following statements hold.*

1. *If $\hat{x} \in \mathcal{X}$ is the unique optimal solution to $\mathcal{P}(\mathcal{U})^{l}_{\lambda}$ for some $\lambda \in \mathbb{R}^k_{\geq}$, then $\hat{x}$ is lower set less ordered strictly efficient solution to $\mathcal{P}(\mathcal{U})$.*

2. *If $\hat{x} \in \mathcal{X}$ is an optimal solution to $\mathcal{P}(\mathcal{U})^l_\lambda$ for some $\lambda \in \mathbb{R}^k_{\{>,\geq\}}$ and*

$$\min_{\xi \in \mathcal{U}} \sum_{i=1}^{k} \lambda_i f_i(x, \xi)$$

exists for all $x \in \mathcal{X}$, then $\hat{x}$ is lower set less ordered [·/weakly] efficient solution to $\mathcal{P}(\mathcal{U})$.

Remark 3.4. To prove Theorem 3.3, we can use a proof with similar reasoning as for Theorem 3.1, only with the assumption that $\hat{x}$ is lower set less ordered [strictly/·/weakly] efficient.

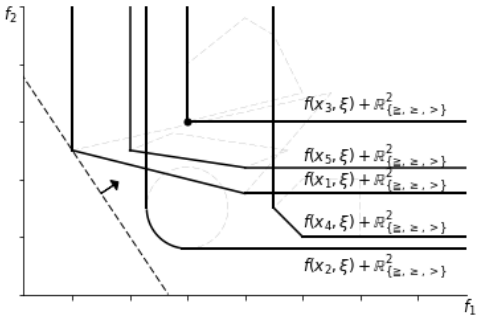


Figure 3.10: Examples of weights to find x_1 as lower set less strictly efficient. Here $\lambda = [2/5, 3/5]$.

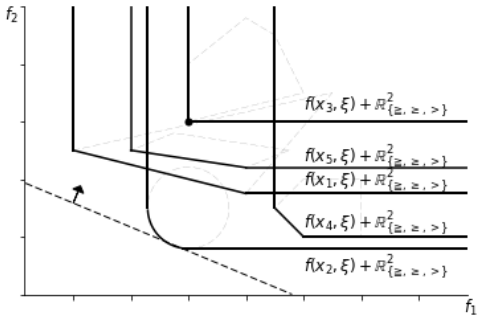


Figure 3.11: Examples of weights to find x_2 as lower set less strictly efficient. Here $\lambda = [5/8, 3/8]$.

ϵ-constraint scalarization

The way we extend the framework from $\mathcal{P}_{(\epsilon,i)}$ to use this method in order for computing lower set less ordered efficient solutions is to insert the problem formulation obtained in 3.2.1, $\mathcal{P}(\mathcal{U})^{\mathrm{l}}$:

$$\begin{aligned} \min_{x \in \mathcal{X}} \inf_{\xi \in \mathcal{U}} \quad & f_i(x, \xi) \\ \textbf{s.t.} \inf_{\xi \in \mathcal{U}} \quad & f_i(x, \xi) \leq \epsilon_j \, \forall \, j \neq i. \end{aligned} \qquad \mathcal{P}(\mathcal{U})^{\mathrm{l}}_{(\epsilon,i)}$$

Given a set of vectors $\mathcal{E}$, we can now compute lower set less ordered efficient solutions by solving $\mathcal{P}(\mathcal{U})^{\mathrm{l}}_{(\epsilon,i)}$ for each $i \in \{1, ..., k\}$ and every $\epsilon \in \mathcal{E}$.

Theorem 3.4. *Theorem 26 Köbis [10] Given an uncertain multi-objective optimization problem $\mathcal{P}(\mathcal{U})$, the following statements hold.*

1. *If $\hat{x} \in \mathcal{X}$ is the unique optimal solution to $\mathcal{P}(\mathcal{U})^l_{(\epsilon,i)}$ for some $\epsilon \in \mathbb{R}^k$ and some $i \in \{1, .., k\}$, then $\hat{x}$ is lower set less strictly efficient solution to $\mathcal{P}(\mathcal{U})$.*

2. *If $\hat{x} \in \mathcal{X}$ is an optimal solution to $\mathcal{P}(\mathcal{U})^l_{(\epsilon,i)}$ for some $\epsilon \in \mathbb{R}^k$ and some $i \in \{1,..,k\}$ and*

$$\min_{\xi \in \mathcal{U}} f_i(x,\xi)$$

exists for all $x \in \mathcal{X}$, then $\hat{x}$ is lower set less weakly efficient solution to $\mathcal{P}(\mathcal{U})$.

Remark 3.5. To prove Theorem 3.4, we can use a proof that will look similar to the one used for Theorem 3.2, only with the assumption that $\hat{x}$ is lower set less ordered [strictly/·/weakly] efficient.

3.3 Set less order relation

3.3.1 Description and problem formulation

As our first two set order relations were pessimistic and optimistic, it is then natural to define set order relations where we combine the two to obtain a sort of compromise between the two opposites. The first one is the set less order relation which Young [2] introduced

Definition 3.5. A set $A \subseteq \mathbb{R}^k$ *dominates* a set $B \subseteq \mathbb{R}^k$ with respect to the set less order relation, denoted $A \preceq^s_{[\leqq,\leq,<]} B$, with respect to $\mathbb{R}^k_{[\leqq,\leq,<]}$ if

$$A \preceq^s_{[\leqq,\leq,<]} B \iff A \preceq^l_{[\leqq,\leq,<]} B \wedge A \preceq^u_{[\leqq,\leq,<]} B.$$

Remark 3.6. The relation can be equivalently be written as

$$A \preceq^s_{[\leqq,\leq,<]} B \iff (\forall b \in B \exists a \in A : a[\leqq,\leq,<]b) \wedge (\forall a \in A \exists b \in B : a[\leqq,\leq,<]b).$$

This time, the way to frame the problem to obtain this property is to take both the infimum and the supremum respectively of the uncertainty set

$$\min_{x \in \mathcal{X}} \begin{pmatrix} \inf_{\xi \in \mathcal{U}} & f(x,\xi) \\ \sup_{\xi \in \mathcal{U}} & f(x,\xi) \end{pmatrix}. \tag{$\mathcal{P}(\mathcal{U})^{\mathrm{s}}$}$$

3.3.2 Efficiency and interpretation

Definition 3.6. Given an uncertain multi-objective optimization problem $\mathcal{P}(\mathcal{U})$, a solution $x \in \mathcal{X}$ to $\mathcal{P}(\mathcal{U})$ is *set less ordered* [*strictly*/ $\cdot$ / *weakly*] *efficient* if there is no $\bar{x} \in \mathcal{X} \setminus \{x\}$ s.t. $f_{\mathcal{U}}(\bar{x}) \preceq^s_{[\leqq,\leq,<]} f_{\mathcal{U}}(x)$ with respect to $\mathbb{R}^k_{[\leqq,\leq,<]}$, or equivalently written

$$\nexists \bar{x} \in \mathcal{X} \setminus \{x\} : f_{\mathcal{U}}(\bar{x}) + \mathbb{R}^k_{[\leqq,\leq,<]} \supseteq f_{\mathcal{U}}(x) \wedge f_{\mathcal{U}}(\bar{x}) \subseteq f_{\mathcal{U}}(x) - \mathbb{R}^k_{[\leqq,\leq,<]}.$$

We can look at this relation as a middle ground compared to the two it is a mixture of - the efficient solutions for the previous two form a subset of the efficient solutions in this relation. So if the solution is either upper or lower set less ordered efficient, it is efficient for this relation. However, it is

important to notice that in this relation, a solution is efficient if not any other solution dominates it in *both* the worst and best case. Therefore, it might be possible that a solution is efficient for this relation without being efficient for the two other relations. This is shown in Example 3.1. As a result of the relationship between this relation and the other two mentioned, we know that since x_2 and x_4 is upper set less ordered strictly efficient and x_1 and x_2 is lower set less ordered strictly efficient these three are also automatically set less ordered strictly efficient. However, we also have another feasible solution that is neither but still set less ordered strictly efficient. If we look more closely at Figures 3.3 and 3.9, we observe that only x_2 dominate x_5's worst-case scenario while only x_1 dominate x_5's best-case scenario respectively. That said, none of the solutions dominate in both relations. Hence, x_5 is set less ordered strictly efficient even though it is neither upper set less ordered strictly efficient nor lower set less ordered strictly efficient.

3.3.3 Computing set less ordered efficient solutions

To compute set less ordered efficient solutions, we can use the same extension of framework as for the former two ordered efficient solutions in 3.1.3 and 3.2.3.

Weighted sum scalarization

The way we extend the framework from $\mathcal{P}_\lambda$ to use this method to compute set less ordered strictly efficient solutions is to insert the problem formulation obtained in 3.3.1, $\mathcal{P}(\mathcal{U})^{\text{s}}$:

$$\min_{x\in\mathcal{X}} \quad \begin{pmatrix} \inf_{\xi\in\mathcal{U}} & \sum_{i=1}^{k} \lambda_i f_i(x,\xi) \\ \sup_{\xi\in\mathcal{U}} & \sum_{i=1}^{k} \lambda_i f_i(x,\xi) \end{pmatrix} \qquad \mathcal{P}(\mathcal{U})^{\text{s}}_{\lambda}$$

Given a set of scalarization vectors Λ, we can now compute set less ordered efficient solutions by solving $\mathcal{P}(\mathcal{U})^{\text{s}}_{\lambda}$ for every $\lambda \in \Lambda$. As discussed in 3.3.2, x_5 is a set less ordered strictly efficient solution even though it is neither upper nor lower set ordered strictly efficient.

Theorem 3.5. ***(Theorem 23 Ide et al. [9])*** *Given an uncertain multi-objective optimization problem $\mathcal{P}(\mathcal{U})$, the following statements hold.*

1. *If $\hat{x} \in \mathcal{X}$ is strictly efficient to $\mathcal{P}(\mathcal{U})^{s}_{\lambda}$ for some $\lambda \in \mathbb{R}^{k}_{\geq}$, then $\hat{x}$ is set less ordered strictly efficient solution to $\mathcal{P}(\mathcal{U})$.*

2. *If $\hat{x} \in \mathcal{X}$ is weakly efficient to $\mathcal{P}(\mathcal{U})^s_\lambda$ for some $\lambda \in \mathbb{R}^k_{\{>,\geq\}}$ and*

$$\min_{\xi \in \mathcal{U}} \sum_{i=1}^{k} \lambda_i f_i(x,\xi) \text{ and } \max_{\xi \in \mathcal{U}} \sum_{i=1}^{k} \lambda_i f_i(x,\xi)$$

exists for all $x \in \mathcal{X}$, then $\hat{x}$ is set less ordered [·/weakly] efficient solution to $\mathcal{P}(\mathcal{U})$.

Remark 3.7. To prove Theorem 3.5, we use a similar proof as for Theorem 3.1, but with the assumption that $\hat{x}$ is both lower set less ordered [strictly/·/weakly] efficient *and* upper set less ordered [strictly/·/weakly] efficient and then use both these two facts in the proof.

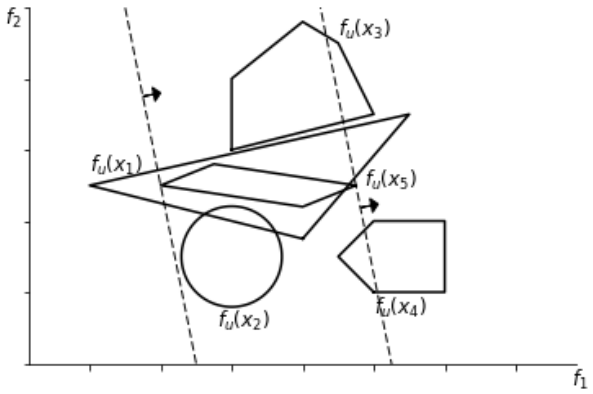


Figure 3.12: Examples of weights to find x_5 as set less strictly efficient. Here $\lambda = [1/6, 5/6]$.

As we can observe in Figure 3.12 that there exist weights where only x_1 is better than x_5 for the best case, and only x_2 is better than x_5 in the worst case. Hence, since none of the solutions are better than x_5 in both cases given the weights, x_5 is here found to be set less ordered strictly efficient using the weighted sum scalarization method.

ϵ-constraint scalarization

The way we extend the framework from $\mathcal{P}_{(\epsilon,i)}$ to use this method to compute set less ordered efficient solutions is to insert the problem formulation obtained in 3.3.1, $\mathcal{P}(\mathcal{U})^{\text{s}}$:

$$\begin{aligned} \min_{x\in\mathcal{X}} \quad & \begin{pmatrix} \inf_{\xi\in\mathcal{U}} & \lambda_i f_i(x,\xi) \\ \sup_{\xi\in\mathcal{U}} & \lambda_i f_i(x,\xi) \end{pmatrix} \\ \textbf{s.t.} \quad & \inf_{\xi\in\mathcal{U}} \quad f_j(x,\xi) \leq \epsilon_j \forall\, j \neq i \qquad\qquad \mathcal{P}(\mathcal{U})^{\mathrm{s}}_{(\epsilon,i)} \\ & \sup_{\xi\in\mathcal{U}} \quad f_j(x,\xi) \leq \epsilon_j \forall\, j \neq i. \end{aligned}$$

Given a set of vectors $\mathcal{E}$, we can now compute lower set less ordered efficient solutions by solving $\mathcal{P}(\mathcal{U})^{\mathrm{s}}_{(\epsilon,i)}$ for each $i \in \{1,...,k\}$ and every $\epsilon \in \mathcal{E}$.

Theorem 3.6. *Given an uncertain multi-objective optimization problem $\mathcal{P}(\mathcal{U})$, the following statements hold.*

1. *(a) If $\hat{x} \in \mathcal{X}$ is strictly efficient to $\mathcal{P}(\mathcal{U})^{s}_{(\epsilon,i)}$ for some $\epsilon \in \mathbb{R}^k$ and some $i \in \{1,..,k\}$, then $\hat{x}$ is set less strictly efficient solution to $\mathcal{P}(\mathcal{U})$.*
2. *(b) If $\hat{x} \in \mathcal{X}$ is weakly efficient to $\mathcal{P}(\mathcal{U})^{s}_{(\epsilon,i)}$ for some $\epsilon \in \mathbb{R}^k$ and some $i \in \{1,..,k\}$ and $\max_{\xi\in\mathcal{U}} f_i(x,\xi)$ exists for all $x \in \mathcal{X}$, then $\hat{x}$ is set less weakly efficient solution to $\mathcal{P}(\mathcal{U})$.*

Remark 3.8. The proof we used to prove Theorem 3.6 is similar to the one we used to prove Theorem 3.2, but instead with the assumption that $\hat{x}$ is both lower set less ordered [strictly/·/weakly] efficient *and* upper set less ordered [strictly/·/weakly] efficient and then combine those two.

3.4 Strict set less order relation

3.4.1 Description

The other relation is the strict set less order relation, which was first introduced in Ide et al. [3] as alternative set less order relation.

Definition 3.7. A set $A \subseteq \mathbb{R}^k$ *dominates* a set $B \subseteq \mathbb{R}^k$ with respect to the strict set less order relation, denoted $A \preceq^{ss}_{[\leqq,\leq,<]} B$, with respect to $\mathbb{R}^k_{[\leqq,\leq,<]}$ if

$$A \preceq^{ss}_{[\leqq,\leq,<]} B \iff A \preceq^{l}_{[\leqq,\leq,<]} B \vee A \preceq^{u}_{[\leqq,\leq,<]} B.$$

Remark 3.9. The relation can equivalently be written as

$$A \preceq^{ss}_{[\leqq,\leq,<]} B \iff (\forall b \in B \exists a \in A : a [\leqq, \leq, <] b) \vee (\forall a \in A \exists b \in B : a [\leqq, \leq, <] b)$$

3.4.2 Efficiency and interpretation

Definition 3.8. Given an uncertain multi-objective optimization problem $\mathcal{P}(\mathcal{U})$, a solution $x \in \mathcal{X}$ to $\mathcal{P}(\mathcal{U})$ is *strict set less ordered* [*strictly*/ $\cdot$ / *weakly*] *efficient* if there is no $\bar{x} \in \mathcal{X} \setminus \{x\}$ s.t. $f_{\mathcal{U}}(\bar{x}) \preceq^{ss}_{[\leqq,\leq,<]} f_{\mathcal{U}}(x)$ with respect to $\mathbb{R}^k_{[\leqq,\leq,<]}$, or equivalently written

$$\nexists \bar{x} \in \mathcal{X} \setminus \{x\} : f_{\mathcal{U}}(\bar{x}) + \mathbb{R}^k_{[\leqq,\leq,<]} \supseteq f_{\mathcal{U}}(x) \vee f_{\mathcal{U}}(\bar{x}) \subseteq f_{\mathcal{U}}(x) - \mathbb{R}^k_{[\leqq,\leq,<]}.$$

For a solution set $f_{\mathcal{U}}(x)$ to be efficient in this relation, there can not exist another solution set $f_{\mathcal{U}}(\bar{x})$ such that neither the best nor worst-case scenario for $\bar{x}$ is lower than the best or worst-case scenario for x. This is a strict property; hence this solution set might be sparse or even empty for some problems. To see this, we notice that to be efficient in this relation, the solution needs to be *both* upper and lower set less ordered efficient. Hence, this solution set can be seen as really good as it is the top in both the worst case and the best case compared to the other feasible solutions. If we look at Example 3.1, we actually have such a solution, x_2. It is both upper and lower set less ordered strictly efficient and hence also strict set less ordered strictly efficient.

3.4.3 Computing strict set less ordered efficient solutions

To obtain strict set less ordered efficient solutions, we use the efficient solutions computed in 3.1.3 and 3.2.3 and find the intersection of the two solution sets. The resulting set is the set of strict set less ordered efficient solutions.

3.5 Relationship between the solutions in the different relations

In this chapter, we have introduced several set order relations. During the different sections, we have been discussing the relationship between a few of the relations. In this section, we want to illustrate how the different relations are related to each other.

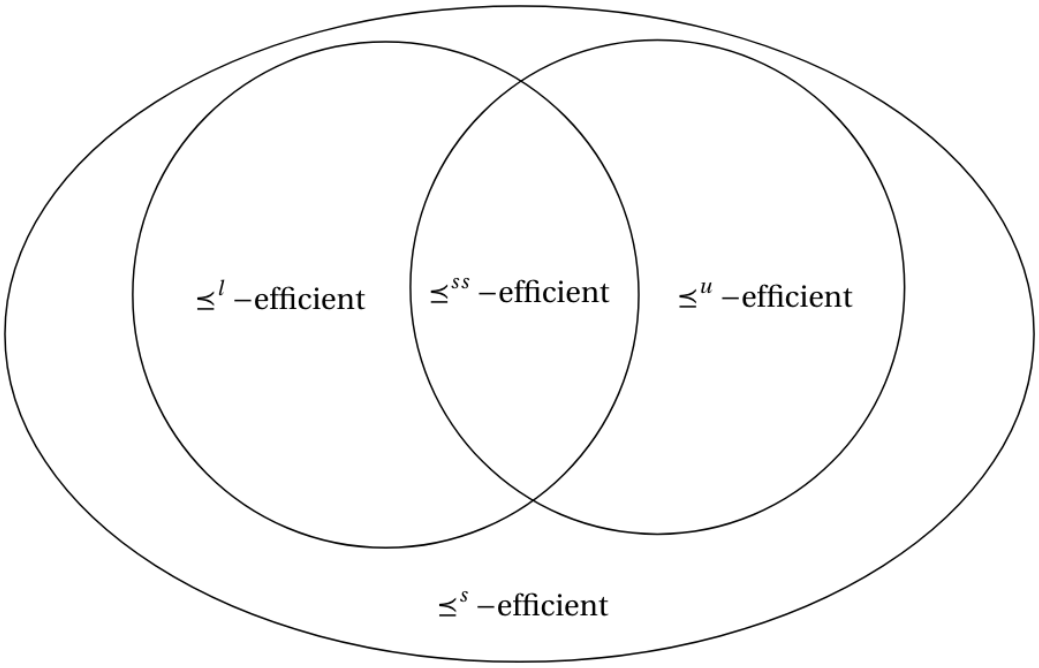


Figure 3.13: Venn diagram of the different set order relations introduced in the chapter.

We begin with the least strict relation, the set less order relation. The solution set of this relation is the union of the lower and upper set less ordered efficient solutions. In addition to this, as discussed in 3.3.1, it is possible for a solution to be neither lower nor upper set ordered strictly efficient but still be set less ordered strictly efficient solution. Next, we have the solutions that are lower and upper set less ordered efficient. These are both a subset but not necessarily equal to the set less ordered efficient solutions. Lastly, we have the strict set less ordered efficient solutions. These solutions are the intersection of the lower and upper set less ordered efficient solutions. A strict set less ordered efficient solution needs to be both a lower and upper set less ordered efficient solution.

Chapter 4

Non-convex feasible sets

We want to have methods that can find all [upper/lower/·/strict] set less ordered solutions given any uncertain multi-objective optimization problem. As we saw with Example 3.2, ϵ-constraint method is not able to fulfill this. It is also well known that for problems where the feasible set is non-convex, the weighted sum scalarization method can not always find all efficient solutions. We will illustrate this with several examples, which we are going to look at throughout this chapter. Therefore, we want to introduce methods to solve problems when the feasible set is not convex. These frameworks can also solve problems with disconnected feasible sets. In this chapter, we are going to focus on introducing two methods. These approaches have primarily been used in multi-objective optimization problems, not uncertain. Hence, we are going to look at the deterministic version while we get to know the methods. In further work, we want to see if we can use them to obtain all efficient solutions for our different set order relations for uncertain multi-objective optimization. So given our already introduced problem formulation ($\mathcal{P}$), we present the first example of the chapter

Example 4.1. Let $\mathcal{X} = \{x \in \mathbb{R}^2_{\geqq} : (x_1-1)^2 + (x_2-1)^2 - 1 \leq 0, 1 - x_1^2 - x_2^2 \leq 0\}$ and $f_1(x) = x_1, f_2(x) = x_2$. Then the efficient solutions is given by $\mathcal{X}_E = \{x \in \mathcal{X} : 1 - x_1^2 - x_2^2 = 0\}$.

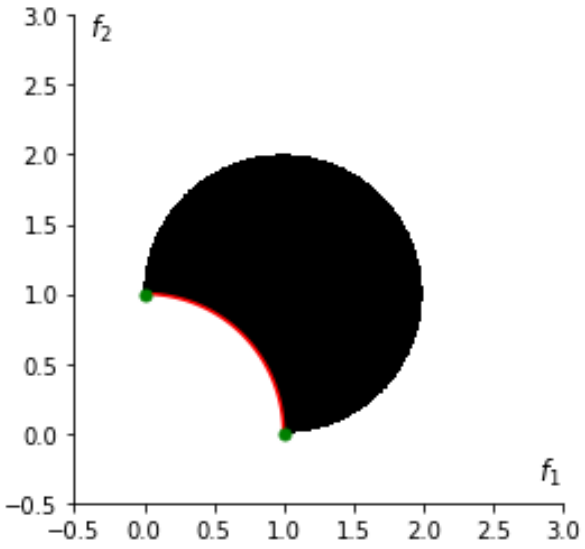


Figure 4.1: Feasible set and Pareto set for Example 4.1 visualized. Feasible set is black while Pareto set is red. The points that are obtainable using weighted sum approach, $\{(1,0),(0,1)\}$, is highlighted in green.

For Example 4.1, the only points which are possible to obtain as efficient solutions with the weighted sum scalarization approach, $\mathcal{P}_\lambda$, are $\{(1,0),(0,1)\}$. The rest of $\mathcal{X}_E$ is unobtainable using this method.

We also want to discuss an example where the feasible set is disconnected. We already know that the weighted sum method can not find the whole Pareto set, the efficient solutions, for all problems. Still, it is an excellent illustration of how powerful the method we will introduce in this chapter is.

Example 4.2. Let $\mathcal{X} = \{x \in \mathbb{R}^2_{\geqq} : (x_1-0.5)^2+(x_2-0.5)^2-1 \leq 0, x_1^2+x_2^2-2 \geq 0, x_1^2+x_2^2-2x_1x_2-0.05 \geq 0\}$ and $f_1(x) = x_1$, $f_2(x) = x_2$. Then the efficient solutions are given by $\mathcal{X}_E = \{x \in \mathcal{X} : x_1^2+x_2^2-2 = 0\}$.

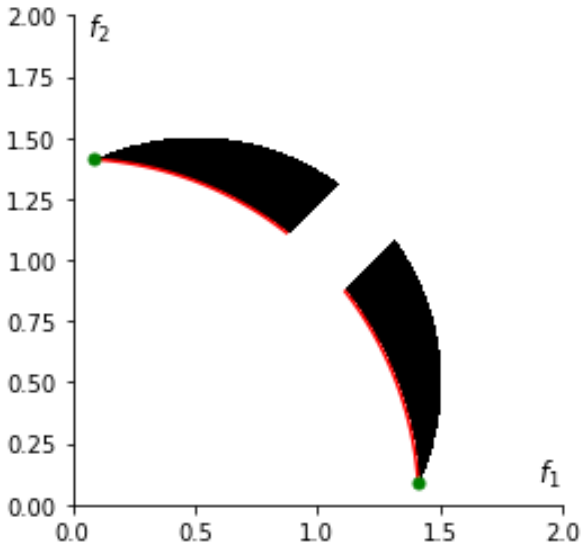


Figure 4.2: Illustration of both the feasible set and Pareto set for Example 4.2. Feasible set is shown as black while Pareto set is in red. The two points that are obtainable using weighted sum approach, $\{(\frac{3+\sqrt{7}}{4}, \frac{3-\sqrt{7}}{4}), (\frac{3-\sqrt{7}}{4}, \frac{3+\sqrt{7}}{4})\}$, are set in green.

For Example 4.2, the only points which are possible to obtain as efficient solutions with the weighted sum method $\mathcal{P}_\lambda$ are $\{(\frac{3+\sqrt{7}}{4}, \frac{3-\sqrt{7}}{4}), (\frac{3-\sqrt{7}}{4}, \frac{3+\sqrt{7}}{4})\}$. The rest of $\mathcal{X}_E$ is unobtainable using this method.

To deal with both these examples and find the whole Pareto set for both of them, we introduce two methods not previously discussed in this paper.

4.1 Weighted-Constraint Method

This method was first introduced by Burachik et al. [4].

4.1.1 Description

Given the different objective functions $f_i,\ i = 1,...,k$ and some positive weights $w \in W^{++} := \{w \in \mathbb{R}^k | w_i > 0, \sum_{i=1}^k w_i = 1\}$, we consider the problem

$$\begin{aligned} \min_{x\in\mathcal{X}} \quad & w_d f_d(x) \\ \textbf{s.t.} \quad & w_i f_i(x) \leq w_d f_d(x), i = 1,...,k, i \neq d, \end{aligned} \qquad \mathcal{P}_{(w,d)}$$

which we refer to as the d^{th}-objective weighted constraint problem. Then for a fixed d and w we define the feasible set as

$$\mathcal{X}_w^d := \{x \in \mathcal{X} | w_i f_i(x) \leq w_d f_d(x), \forall\, i \neq d\}$$

and the solution set of $(\mathcal{P}_{(w,d)})$ as

$$\mathcal{S}_w^d := \{x \in \mathcal{X} | x \text{ solves } (\mathcal{P}_{(w,d)})\}.$$

We also define

$$W(x) := \{w \in W^{++} | x \in \mathcal{S}_w^d, \forall\, d = 1,...,k\}.$$

It is possible that $W(x) = \emptyset$ for some $x \in \mathcal{X}$. To now generate an approximation of the Pareto front, we just solve $\mathcal{P}_{(w,d)}$ for all $d \in \{1,...,k\}$ over a grid of values w. Then for all $w' \in W^{++}$, we have

$$\bigcap_{d=1}^{k} \mathcal{S}_{w'}^d \subseteq \text{WE}(\mathcal{P}).$$

Where we have denoted $\text{WE}(\mathcal{P})$ as the weak efficient solutions of $\mathcal{P}$. This relationship thus shows a way to compute weak efficient solutions by solving $\mathcal{P}_{(w,d)}$ for all $d = 1,...,k$ for some $w' \in W^{++}$. If $\bigcap_{d=1}^k \mathcal{S}_{w'}^d \neq \emptyset$ then we have obtained at least one weak efficient solution.

Theorem 4.1. ***(Theorem 3.1. Burachik et al. [4])*** *$\hat{x} \in \mathcal{X}$ is a weak efficient solution of $\mathcal{P}$ $\iff$ there exist some $w \in W^{++}$ such that $\hat{x}$ solves $\mathcal{P}_{(w,d)}$ for all $d \in \{1,...,k\}$.*

Proof. $\implies$ Assume $\hat{x} \in \mathcal{X}$ is a weak efficient solution to $\mathcal{P}$. Without loss of generality we say that $f_i(x) > 0, i = 1,...,k \forall x \in \mathcal{X}$. Then we define

$$w_i := \frac{1/f_i(\hat{x})}{\sum_{j=1}^{k} 1/f_j(\hat{x})}.$$

For this choice, $w \in W^{++}$ and $\hat{x}$ satisfies all constraints as equalities, or in other words

$$w_i f_i(\hat{x}) = w_d f_d(\hat{x}), i = 1,..,k, i \neq d. \tag{1}$$

If $\hat{x}$ is not a solution of $\mathcal{P}_{(w,d)}$ for some d, then there exist $\bar{x} \in \mathcal{X}$ such that

$$w_d f_d(\bar{x}) < w_d f_d(\hat{x}) \tag{2}$$

and

$$w_i f_i(\bar{x}) \leq w_d f_d(\bar{x}), i = 1,...,k, i \neq d.$$

Hence, we can write

$$w_i f_i(\bar{x}) \leq w_d f_d(\bar{x}) < w_d f_d(\hat{x}), i = 1,...,k, i \neq d.$$

Then, by (1), we can write

$$w_i f_i(\bar{x}) \leq w_d f_d(\bar{x}) < w_i f_i(\hat{x}), i = 1,...,k, i \neq d, \tag{3}$$

and since $w_i > 0$, if we combine (2) with (3) we get

$$f_i(\bar{x}) < f_i(\hat{x}), i = 1, ..., k.$$

This contradicts the weak efficiency of $\hat{x}$.

$\Longleftarrow$ Assume that $w \in W^{++}$ is such that $\bar{x}$ solves $\mathcal{P}_{(w,d)}$ for all d. Suppose that $\bar{x} \in \mathcal{X}$ is not a weak efficient point of $\mathcal{P}$. Then there must exist $\hat{x} \in \mathcal{X} \setminus \bar{x}$ such that

$$f_i(\hat{x}) < f_i(\bar{x}), i = 1, ..., k. \tag{4}$$

Then there must exist a d such that $\hat{x} \in \mathcal{X}_w^d$. Therefore, from (4) we can write $w_d f_d(\hat{x}) < w_d f_d(\bar{x})$ where $w_d > 0$. This contradicts that $\bar{x}$ solves $\mathcal{P}_{(w,d)}$. □

Remark 4.1. **(Remark 3.1. Burachik et al. [4])** The $\Longrightarrow$ part of Theorem 4.1 holds for efficient solutions since every efficient solution is a weak efficient solution. However, if a point solves $\mathcal{P}_{(w,d)}$ for all $d \in 1, ..., k$, then this does not necessarily imply that the solution is efficient unless all objective functions are strictly convex.

4.1.2 Computation of efficient points and interpretation

We will now illustrate how the method finds efficient solutions in examples 3.1 and 3.2.

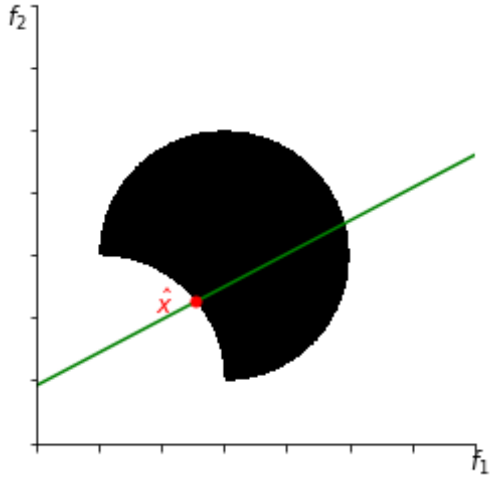


Figure 4.3: Example of obtaining an efficient solution $\hat{x} = (\frac{11}{\sqrt{202}}, \frac{9}{\sqrt{202}})$ for a given $w = [\frac{9}{20}, \frac{11}{20}]$. The green line shows the line $w_1 x_1 = w_2 x_2$.

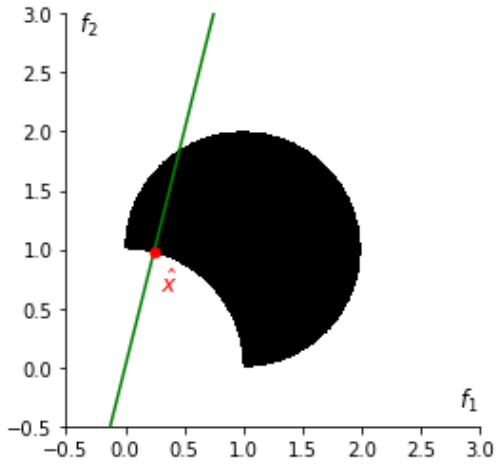


Figure 4.4: Example of obtaining an efficient solution $\hat{x} = (\frac{1}{\sqrt{17}}, \frac{4}{\sqrt{17}})$ for a given $w = [\frac{4}{5}, \frac{1}{5}]$. The green line shows the line $w_1 x_1 = w_2 x_2$.

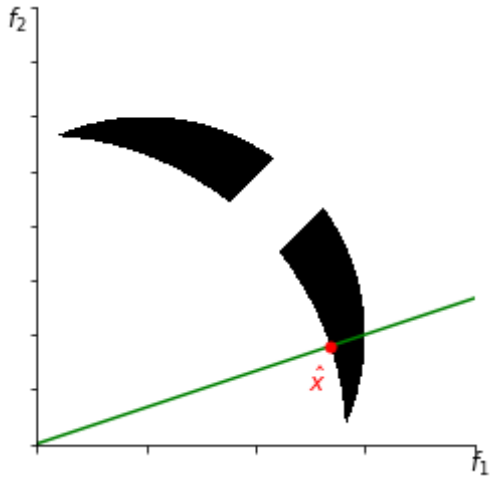


Figure 4.5: Example of obtaining an efficient solution $\hat{x} = (\frac{3}{\sqrt{5}}, \frac{1}{\sqrt{5}})$ for a given $w = [\frac{1}{4}, \frac{3}{4}]$. The green line shows the line $w_1x_1 = w_2x_2$.

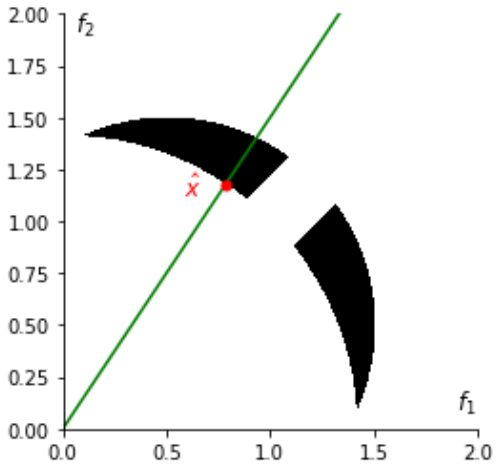


Figure 4.6: Example of obtaining an efficient solution $\hat{x} = (\sqrt{\frac{8}{13}}, \frac{3}{2}\sqrt{\frac{8}{13}})$ for a given $w = [\frac{3}{5}, \frac{2}{5}]$. The green line shows the line $w_1x_1 = w_2x_2$.

If we look at $\mathcal{P}_{(w,d)}$ and combine that formulation with the definition of an efficient solution for the method, $\bigcap_{d=1}^{k} \mathcal{S}_{w'}^{d}$, we see that the only set the efficient solution can lie on in these examples for a given w' is the line defined by $w_1x_1 = w_2x_2$. This can be explained by the fact that for $\mathcal{P}_{(w',1)}$, the constraint is given by $w_2x_2 - w_1x_1 \leq 0$, while for $\mathcal{P}_{(w',2)}$, the constraint is given by $w_1x_1 - w_2x_2 \leq 0$. The points $x \in \mathcal{X}$ eligible for being an efficient solution for that given w' are therefore the set of points which fulfill the equation $w_1x_1 = w_2x_2$. Hence, by choosing a range of $w \in W^{++}$, it is possible to estimate the Pareto front. However, it is important to note that not all w' will produce an efficient solution. In Figure 4.3 and Figure 4.4, two different w's have been chosen to find two of the solutions $\hat{x}$ that make up the efficient solutions in Example 3.1, while in Figure 4.5 and Figure 4.6, two different w's have been chosen to find two of the solutions $\hat{x}$ that make up the efficient solutions in Example 3.2.

One way one can compute an estimation of the Pareto front in the two examples using this method is first to find the $\hat{x}$ which optimize the different objective functions isolated. For Example 3.1 and 3.2 these points are illustrated in 4.1 and 4.2 in green. By finding these points, we know that the rest of the Pareto front is in-between the weights used to obtain these two endpoints of the set. By then choosing how many points we want to approximate the set, choose the increment in weights we want. If we set the weight w_{f_1} at the weight used in finding the point optimizing the first objective function and w_{f_2} at the weight used in finding the point

optimizing the second objective function, we can now approximate the Pareto front by setting the weights as $w_n = w_{f_1} + \frac{n(w_{f_2} - w_{f_1})}{N}, n = 1, ..., N-1$. By doing this, we get an approximation of the Pareto front with N points, given that all optimization problems have a solution. For Example 3.1 this is true, while in Example 3.2 we can observe that some of the weight might be set such that $\bigcap_{d=1}^{k} \mathcal{S}_{w'}^{d} = \emptyset$. Then the approximation will contain fewer points if one does not modify the algorithm not to divide the interval between w_{f_1} and w_{f_2} equally to avoid this. The problem with trying to fix this issue is that one is unsure which weights will produce an efficient solution.

On the basis of these uplifting results, we now want to see if we can use the weighted constraint method in uncertain multi-objective optimization to find the solutions we have been discussing in earlier chapters. First, we need to prove that solving a problem with the weighted constraint method will produce a solution we are looking for. Then we can compare it to another method we have been looking at already, the weighted sum method, to see if it can perform well in finding the solutions we are looking for.

Chapter 5

Weighted constraint method in uncertain multi-objective optimization

5.1 Formulation

In this chapter we want to find out if the weighted constraint method we introduced in the last chapter (Chapter 4), is able to find all [upper/lower/·/strict] set ordered efficient solutions in uncertain multi-objective optimization problems. So far, when we introduced the method, we only looked at deterministic multi-objective optimization. To find out if the method can be useful in uncertain multi-objective optimization problems, we need to extend the framework from $\mathcal{P}_{(w,d)}$. In this chapter, we are going to focus on computing upper less ordered efficient solutions. Hence, we have to look at the supremum of the feasible elements. Therefore, we consider:

$$\begin{aligned} &\min_{x\in\mathcal{X}} \sup_{\xi\in\mathcal{U}} w_d f_d(x,\xi) \\ &\textbf{s.t.} \sup_{\xi\in\mathcal{U}} w_i f_i(x,\xi) \leq \sup_{\xi\in\mathcal{U}} w_d f_d(x,\xi), i = 1,...,k, i \neq d, \end{aligned} \qquad \mathcal{P}(\mathcal{U})^u_{(w,d)}$$

Remark 5.1. We are only looking at one of the relations in this chapter, the upper set less order relation. The arguments and results will be similar for the other relations, so we will not focus on these three in this chapter.

5.2 Closer look at the weighted constraint method

Before we can use the weighted constraint method on uncertain multi-objective optimization problems, we need to find out if we obtain the solutions we are looking for using the weighted constraint method. As mentioned in the last subsection, we will focus on the upper set less order efficient solutions as a proof of concept.

Theorem 5.1. *Given an uncertain multi-objective optimization problem $\mathcal{P}(\mathcal{U})$, the following statements hold.*

1. *If $\hat{x} \in \mathcal{X}$ is the unique optimal solution to $\mathcal{P}(\mathcal{U})^u_{(w,d)}$ for some $w \in \mathbb{R}^k_>$ and all $d \in \{1,..,k\}$, then $\hat{x}$ is upper set less ordered strictly efficient solution to $\mathcal{P}(\mathcal{U})$.*
2. *If $\hat{x} \in \mathcal{X}$ is an optimal solution to $\mathcal{P}(\mathcal{U})^u_{(w,d)}$ for some $w \in \mathbb{R}^k_>$ and all $d \in \{1,..,k\}$ and*

$$\max_{\xi \in \mathcal{U}} f_d(x,\xi)$$

exists for all $x \in \mathcal{X}$, then $\hat{x}$ is upper set less ordered weakly efficient solution to $\mathcal{P}(\mathcal{U})$.

Proof. 1. Assume $\hat{x}$ is not upper set less ordered strictly efficient for $\mathcal{P}(\mathcal{U})$. Then there exists a $x' \in \mathcal{X}$ such that

$$f_{\mathcal{U}}(x') \subseteq f_{\mathcal{U}}(\hat{x}) - \mathbb{R}^k_{[\geqq,\geq,>]} \implies \forall \xi \in \mathcal{U} \exists \eta \in \mathcal{U} : f(x',\xi) \leqq f(\hat{x},\eta)$$

Further, there must exist a $d' \in \{1,...,k\}$ such that

$$\begin{aligned} &\implies \forall \xi \in \mathcal{U} \exists \eta \in \mathcal{U} : w_{d'} f_{d'}(x',\xi) \leqq w_{d'} f_{d'}(\hat{x},\eta) \\ &\implies \forall \xi \in \mathcal{U} : w_{d'} f_{d'}(x',\xi) \leqq \sup_{\eta \in \mathcal{U}} w_{d'} f_{d'}(\hat{x},\eta) \end{aligned}$$

$$\implies \sup_{\xi \in \mathcal{U}} w_{d'} f_{d'}(x',\xi) \leqq \sup_{\eta \in \mathcal{U}} w_{d'} f_{d'}(\hat{x},\eta) \tag{1}$$

Now define

$$\sup_{\xi\in\mathcal{U}} w_{d'} f_{d'}(x',\xi) := \max_{i=1,\dots,k} \sup_{\xi\in\mathcal{U}} w_i f_i(x',\xi),$$

then

$$\sup_{\xi\in\mathcal{U}} w_i f_i(x',\xi) \leqq \sup_{\xi\in\mathcal{U}} w_{d'} f_{d'}(x',\xi), i = 1,\dots k, i \neq d'.$$

Hence, $x' \in X_w^{d'}$. (1) implies

$$\sup_{\xi\in\mathcal{U}} w_{d'} f_{d'}(x',\xi) \leqq \sup_{\eta\in\mathcal{U}} w_{d'} f_{d'}(\hat{x},\xi),$$

in contradiction to $\hat{x}$ being uniquely minimal for all $d \in \{1,\dots,k\}$. □

Remark 5.2. The weighted constraint method does not find all upper set less order efficient solutions. In the following example, an upper set less ordered efficient solution exist, which is not found by the weighted constraint method.

Example 5.1. Consider the following problem $\mathcal{P}(\mathcal{U})$, pictured in Figure 5.1: Let $f : \mathbb{R}^2 \times \mathcal{U} \to \mathbb{R}^2$ be given by $f(x,\xi) = x$, $\mathcal{X} := \{z_1, z_2\}$ where $z_1 := \{(x_1 - 1)^2 + (x_2 - 1)^2 \leq 1\}$ and $z_2 := \{x_1 + x_2 \geq 2, \frac{4}{9}x_1 - x_2 - 1 \geq 0, \frac{9}{4}(x_1 - 1) - x_2 \leq 0\}$. Then neither $f_{\mathcal{U}}(z_1) \preceq^u_{[\geqq,\geq,>]} f_{\mathcal{U}}(z_2)$ nor $f_{\mathcal{U}}(z_2) \preceq^u_{[\geqq,\geq,>]} f_{\mathcal{U}}(z_1)$. Therefore, both z_1 and z_2 are upper set less order strictly efficient solutions. However, since $\sup_{\xi\in\mathcal{U}} f_1(z_2,\xi) = \sup_{\xi\in\mathcal{U}} f_1(z_2,\xi) = 1.8$, the only w where z_2 is feasible for both $d = 1$ and $d = 2$ is $w = [0.5, 0.5]$. Furthermore, we observe that z_1 is the unique optimal solutions to both $\mathcal{P}(\mathcal{U})^u_{([0.5,0.5],1)}$ and $\mathcal{P}(\mathcal{U})^u_{([0.5,0.5],2)}$. Therefore, there does not exist a $w \in \mathbb{R}^k_>$ where z_2 is the unique optimal solution to $\mathcal{P}(\mathcal{U})^u_{(w,d)}$. Hence, the weighted constraint method is not able to find all upper set less order efficient solutions for any given $\mathcal{P}(\mathcal{U})$.

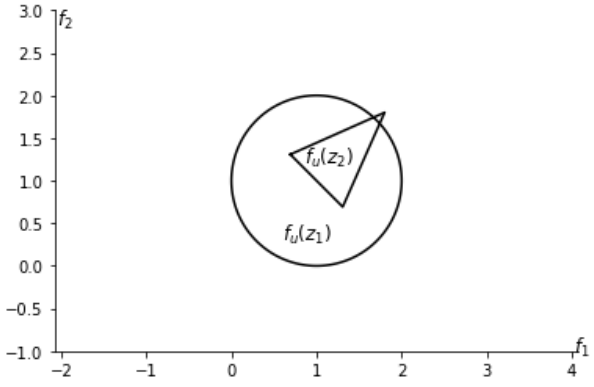


Figure 5.1: Uncertain multi-objective optimization problem with feasible elements $\mathcal{X} = \{z_1, z_2\}$.

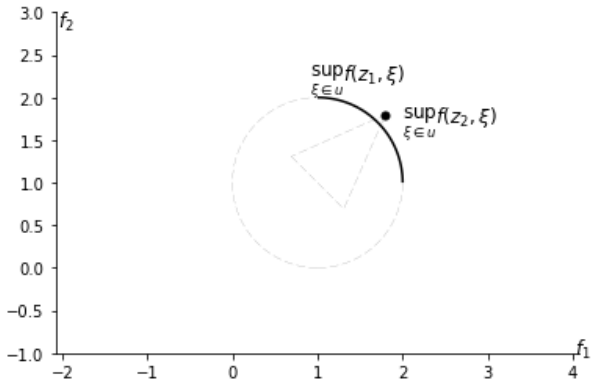


Figure 5.2: Supremum of both feasible elements in example 5.1.

Given this counterexample, we know that there exist problems where the weighted constraint method cannot obtain all upper set less order efficient solutions. Nevertheless, given a set W^{++} of vectors $w \in \mathbb{R}^k_>$, we can now compute upper set less ordered efficient solutions by solving $\mathcal{P}(\mathcal{U})^u_{(w,d)}$ for each $d \in \{1, ..., k\}$ and every $w \in W^{++}$.

5.3 Numerical investigation

Now we know that solutions of $\mathcal{P}(\mathcal{U})^u_{(w,d)}$ will be upper set less ordered efficient solutions (u.s.l.o.e) to our original problem, $\mathcal{P}(\mathcal{U})$. To see how well the method performs in finding these solutions, we want to generate some problem instances and compare the weighted constraint method with a previously discussed method, the weighted sum method. By doing this, we can get some insight into how well the method can perform in finding the solutions we are looking for when working with uncertain multi-objective optimization problems compared to other methods.

Here is the link to the GitHub repository containing my implementation.

5.3.1 General framework

For the numerical results, we want to investigate several aspects when comparing the two methods. We will primarily look at how many upper set less order efficient solutions each method finds and how much time each method uses to obtain the solutions. Since both methods use weights, it is also interesting to see how the methods perform given the precision of the weights. To test all these qualities, we have used randomly generated problem instances of uncertain multi-objective optimization problems.

Parameters

The parameters we can tune to get some results are the number of feasible elements per problem instance, the number of objectives per problem instance, and the number of weights we are using. To explain the number of weights, we give an example. Given a number of weights N, the weights will be on the form $[\frac{N-n}{N}, \frac{n}{N}]$ for $n = 0, 1, ..., N$. So if we have two objectives and $N = 2$, then the set of weights that the methods are going to solve for will be $\{[1,0],[0.5,0.5],[0,1]\}$. The last parameter is the number of problem instances we are creating given the other parameters. For the data to be more insightful, we want to have many problem instances generated for each set of the other parameters. Hence, given the number of weights $N = 2$ and the number of objectives is 2, we could generate 1000 problem instances to find out how the methods perform on average in all these problem instances given the parameters. In this analysis, we will only discuss problem instances with either two or three objective functions. Both methods work for more objective functions, as previously discussed in the

paper, but analyzing two and three are sufficient. Another reason is because of the time spent to solve the problem instances. The number of weights we have to solve for given N is in the order $O(N^{s-1})$, where s is the number of objective functions. Hence, for a given precision, the number of problems needed to solve grows rapidly when given more objective functions.

5.3.2 Two objectives

First, we are going to discuss the performance for when the objective function f has two components. Throughout this section, the two components are given as $f_1(x) = x_1$ and $f_2(x) = x_2$. Further, every feasible element is a set comprised of a circle with a randomly generated center $(x_1, x_2), \{x_1, x_2\} \in [1,4]$ and radius $r \in [0.25, 0.75]$.

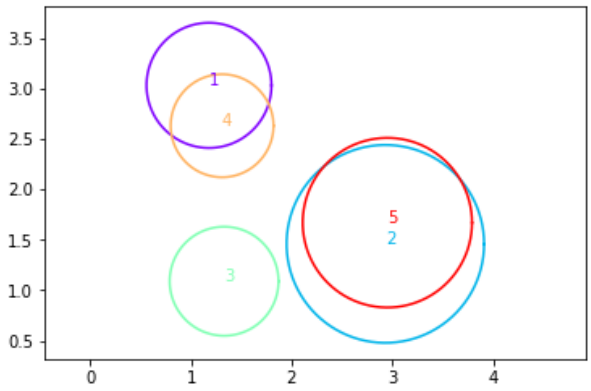


Figure 5.3: Uncertain multi-objective optimization problem with feasible elements $\mathcal{X} = \{1,2,3,4,5\}$.

Figure 5.3 shows an example of one problem instance that we have generated with 5 feasible elements. In this problem instance, elements 1,3 and 4 are upper set less order efficient.

Precision of weights

Given this setup, we wanted to test how the methods perform given the number of weights it is solving for. To check this, we generated problem instances with weights between 2^2 and 2^{15}. It was simulated twice with a different number of feasible elements, 30 and 70 respectively, while the number of problem instances was 1000 both times.

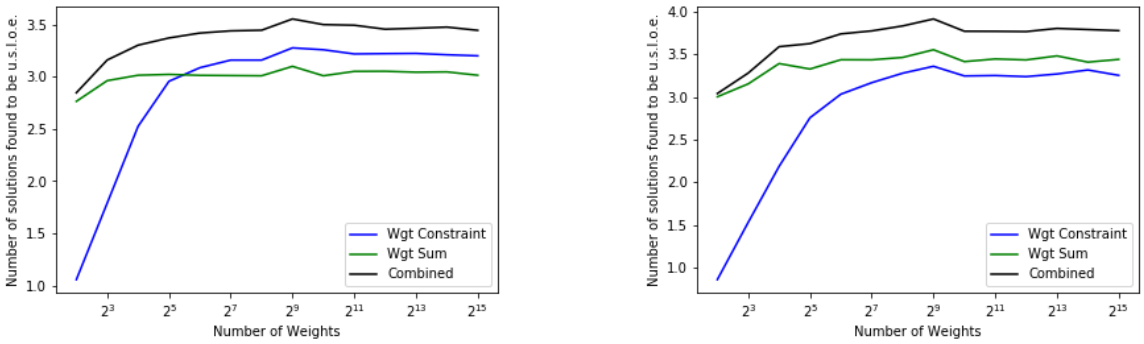


Figure 5.4: Number of upper set less order efficient solutions found per problem instance. Weighted Constraint method in blue, weighted sum method in green and the combined effort of the two in black. The number of feasible elements differs for the two plots. On the left it is 30 feasible elements while it is 70 on the right.

Figure 5.4 is very interesting. Both methods have a point where more precision of weights is irrelevant for the number of feasible elements found to be upper set less order efficient. However, this point is not the same for both methods. This number is much lower for the weighted sum method, 2^4 for 30 feasible elements and 2^6 for 70 feasible elements. It seems like that same point is 2^9 for the weighted constraint method in both plots. This gives an edge to the weighted sum method because it is way faster to run the code with less precision in the weights as the computation time grows in the order of $O(N)$. So if the method can perform the same with a small number of weights as it does with a larger number of weights, then no sacrifice is made to save time running the code.

Number of Feasible Elements

Given the knowledge attained in the last section, the next thing we wanted to test is the number of feasible elements in each problem instance. Therefore, we generated problem instances with feasible elements between 10 and 100. The other parameters are set to $N = 2^9$ as both methods seem to perform at their peak given this precision, and the number of instances was 1000.

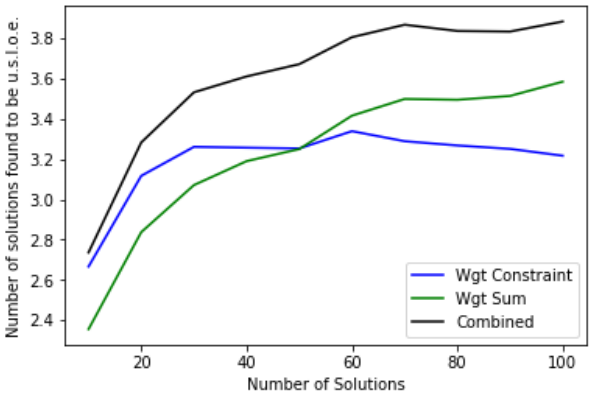


Figure 5.5: Number of upper set less order efficient solutions found per problem instance. Weighted Constraint method in blue, weighted sum method in green, and the combined effort of the two in black.

Figure 5.5 shows that when the number of feasible elements is low, the weighted constraint method outperforms the weighted sum method. However, when the number of feasible elements gets higher than 50, the weighted sum method outperforms the weighted constraint method. This is not what we expected, as the belief was that the weighted constraint method would perform better than the weighted sum method even when the number of feasible elements grows. The reasoning behind this comes from deterministic multi-objective optimization. Because when we are working with those problems, the weighted sum method can only guarantee to find the Pareto front if the set is convex. If it is non-convex, it is possible that the method is not able to obtain the set we are looking for. However, the weighted constraint method does not have this issue. It can find the Pareto front even though the set is non-convex. In addition, the weighted sum method is a linear method, while the weighted constraint method is a non-linear method. Hence, it is unclear why the weighted sum method outperforms the weighted constraint method, but there are some differences. One is that the weighted constraint method is subjected to constraints, while the weighted sum method is not. The other reason is that for the weighted constraint method to find a feasible element to be an upper set less ordered efficient solution, it needs to beat all other feasible elements for both $d = 1$ and $d = 2$ given a weight w. The more feasible elements in the problem instance, the harder this becomes. On the other hand, for the weighted sum method to find a feasible element to be u.s.l.o.e., it only needs to beat out all other feasible elements once for a given weight w. It is worth mentioning that the objectives include both objectives as a linear

combination in the weighted sum method. In contrast, the weighted constraint method only has one of the objectives to optimize at a time.

5.3.3 Three objectives

Now we shift the analysis to instances where the objective function f has three components. Throughout the section, the three components are given as $f_1(x) = x_1$, $f_2(x) = x_2$ and $f_3(x) = x_3$. Further, every feasible element is a set comprised of a sphere with a randomly generated center $(x_1, x_2, x_3), \{x_1, x_2, x_3\} \in [1,4]$ and radius $r \in [0.25, 0.75]$. Because of computation time, the numbers in this section will be smaller, so the results will have more variance than with two objectives. The main idea of including this section is to see if the results from two objectives translate into three or see if the results from two objectives can be an outlier.

Precision of weights

As for two objectives, we want to start by looking at how the methods perform given the number of weights it is solving for. To check this, we generated problem instances with weights between 2^2 and 2^7. The number of feasible elements is set to 10, while the number of instances is 10. As previously mentioned, the computation time is much more expensive since we are dealing with three objectives, and the parameters are adjusted accordingly.

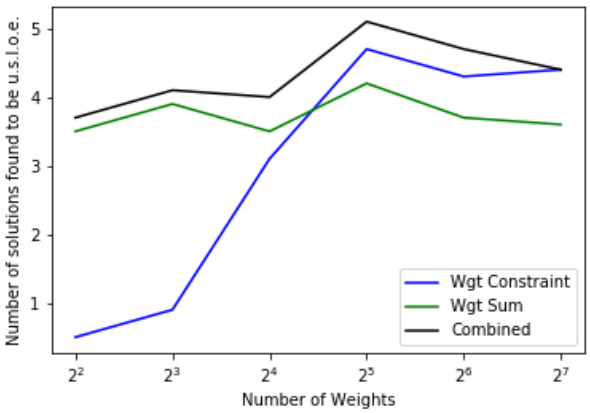


Figure 5.6: Number of upper set less order efficient solutions found per instance. Weighted Constraint method in blue, weighted sum method in green, and the combined effort of the two in black.

Figure 5.6 shows some instability in the graphs. However, the preliminary results agree with the picture given to us in the simulation with two

objectives. The weighted sum method performs almost the same with $N = 2^2$ as it does when $N = 2^8$. On the other hand, the weighted constraint method performs much better given higher precision. Hence, this confirms our first impression from the last section, namely that the weighted sum method can produce the same results given lower accuracy, which is advantageous because then it is faster to run. This still applies for three objectives, giving it an edge as the computation time grows in the order of $O(N^2)$.

Number of feasible elements

We will also look at how the methods perform given a different number of feasible elements with three objectives. Therefore, problem instances with the number of feasible elements between 10 and 100 were generated. In addition to this, we have set the number of weights to $N = 2^6$, and the number of problem instances is 10.

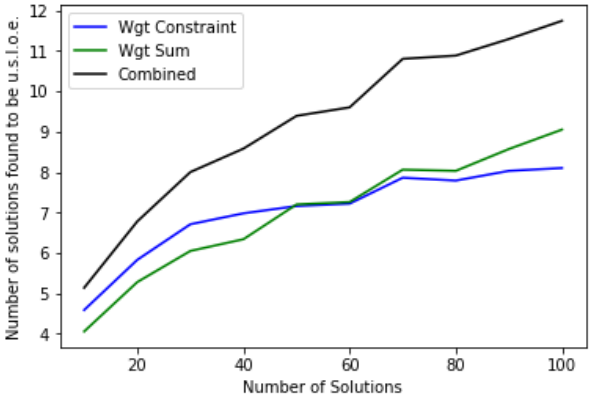


Figure 5.7: Number of upper set less order efficient solutions found per instance. Weighted Constraint method in blue, weighted sum method in green, and the combined effort of the two in black.

Figure 5.7 shows a total agreement with what we saw with two objectives. For problem instances where the number of feasible elements is lower than 50, the weighted constraint method outperforms the weighted sum method. These results indicate that the number of constraints is not the main reason why the weighted constraint method performs worse than the weighted sum method. This is because we see almost the same results as for two objectives, even though the weighted constraint method includes more constraints when we increase the number of objectives. For two objectives, $\mathcal{P}(\mathcal{U})^u_{(w,d)}$ includes one constraint for each d, while this number becomes two when we have three objectives. The weighted sum method overtakes

the performance of the weighted constraint method when the number of feasible elements gets higher than 50 in this simulation as well. Hence, this lowers the probability that the number of constraints is the main problem why the weighted constraint method does not perform at the level of the weighted sum method. Furthermore, this also debunks the other mentioned reason, namely that the explanation for the bad performance is the number of times a feasible element needs to be best for the method to find it as u.s.l.o.e.. If that were the case, it would make more sense for the weighted constraint method to perform worse than the weighted sum method, even for fewer feasible elements. The reason for this is that for a feasible element to be found as u.s.l.o.e. with three objectives, it needs to be best not only for $d = 1$ and $d = 2$ as with two objectives but also for $d = 3$.

5.3.4 Computation time

As discussed in the previous sections, it is not just performance that matters in our comparison. It is also essential to consider computation time.

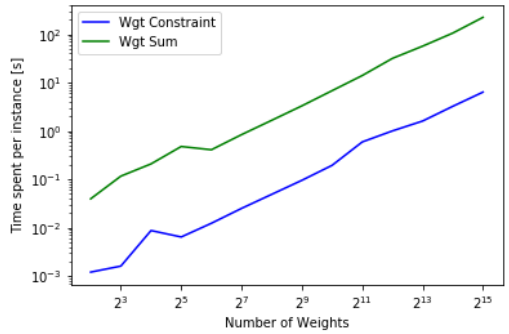


Figure 5.8: Time used per instance given in seconds. Weighted Constraint method in blue and weighted sum method in green.

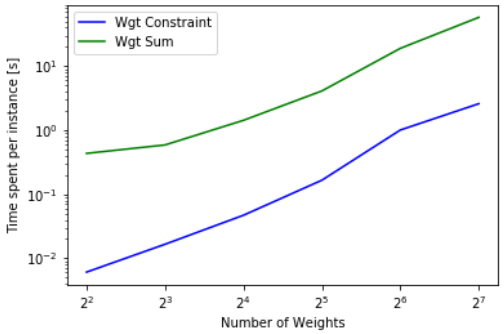


Figure 5.9: Time used per instance given in seconds. Weighted Constraint method in blue and weighted sum method in green.

Both figures were from the simulations when the changing variable was the number of weights, as indicated on the x-axis on both plots. Figure 5.8 and Figure 5.9 show that the weighted constraint method is much faster than the weighted sum method. In both cases, the weighted constraint method is ten times faster per instance it calculates. This information is valuable when assessing the methods against each other. We could attribute this to the characteristics of the sets. Because the weighted constraint method only has one of the objective functions to optimize on, it is well known which points will produce the maximum and the minimum. Hence, it is possible to do a check before each optimization. Either all points are infeasible given the

weights, or the maximum is feasible. In both scenarios, we can skip the computation time of optimizing. In the former scenario, the method can not find a feasible element to be u.s.l.o.e.. For the latter, we quickly found out that the point which produces the maximum is feasible. Hence we choose that point. If none of these two scenarios are true, we have to do the optimization. This knowledge is not something we might be able to use in all problems we are solving, but it shows that the weighted constraint method can have some perks in some problems. The fact that we have some weights where all points are infeasible can be attributed to the constraints that the weighted sum method does not have.

It is crucial to take into consideration the discussion from the last sections. We want to compare the methods on how they perform at their best. Since the weighted sum method does this at a lower precision than the weighted constraint method, the gap in time consumed between the two methods becomes smaller. Hence, the weighted constraint method is not that superior as the plots show at first glance, as the time saved by the weighted constraint method per calculation is negated by the ability of the weighted sum method to perform well with fewer calculations.

5.3.5 Summary

The results in the numerical investigation are two-sided. On one side, the weighted constraint method was much faster to run per calculation than the weighted sum method, which is very positive. However, the edge was not huge because the weighted sum method needs fewer calculations to perform at its best. When looking at the performance of finding upper set less order efficient solutions, the difference was not that clear. For a fewer number of feasible elements, the weighted constraint method outperformed the weighted sum method. This was the case with two and three objectives. When the number of feasible elements increased, the weighted sum method did better.

Chapter 6

Conclusion

This paper first introduced different set order relations for finding different efficient solutions in uncertain multi-objective optimization problems. The upper set less order relation can be connected with risk aversion as the efficient solutions in this relation are not dominated in the worst-case scenario. Hence, the solutions we get from this are a way of hedging against the scenarios where the worst happens. Next, we introduced the lower set less order relation, connected with being risk affine. The efficient solutions in this relation are not dominated in the best-case scenario. Hence, the solutions we get from this have the most significant upside, but on the other hand, we are not safe from these solutions having the worst downsides as well. If we are risk-neutral instead, then the set less order relation might be good for us to use as this is the union of the risk-averse and risk affine solutions. As we saw in examples, it is also possible to have solutions in this relation that are neither upper nor lower set less ordered efficient. These solutions are often the ones that are neither the very best but not the very worst either. The last relation we have introduced is the strict set less order relation. This is the intersection of the risk-averse and the risk affine solutions. As one can tell, these are the best solutions as they have the best upside and are guaranteed not to have the worst outcomes. Depending on the problem at hand, these solutions can be rare, but they are considered the best no matter the risk assessment if they are achievable.

Secondly, we dived into a method used for approximating the Pareto front in multi-objective optimization problems, the weighted constraint method. The method is non-linear, as one of the previously discussed methods, ϵ-constraint is. Since the weighted constraint method can obtain the whole

Pareto set for multi-objective optimization problems, we wanted to check if it could also find the solutions we were looking for in uncertain multi-objective optimization. The last part dealt with this by showing we could use the method to find the solutions we are looking for. Then we compared how the method performed against one of the methods we have previously discussed, the weighted sum method. To do this, we generated uncertain multi-objective optimization problems and made both methods solve them to see how many solutions they could find each. As a proof of concept, we only check for solutions to one of the introduced set orderings, the upper set less order relation. The results showed that the weighted constraint method was much quicker and did better when the problem did not have too many feasible sets. However, when the problem contained many feasible sets, the weighted sum method found more upper set less order efficient solutions. Hence, the weighted constraint method is a viable option, especially for problems with fewer feasible sets. It is also worth emphasizing the importance of saving computational time as the weighted constraint method does when comparing it against the weighted sum method. If we want to use this method to solve problems in real life, we have to remember that companies often have many problems it needs to solve each day. Even though the problems in real life are much more chaotic than those dealt with in this paper, it gives us a feel for how the method can perform against other methods. Hence, the difference we found in computation time can be the difference between useful and useless.

References

[1] D. Kuroiwa. *On natural criteria in set-valued optimization*. RIMS Kokyuroku, 1998. URL: https://repository.kulib.kyoto-u.ac.jp/dspace/handle/2433/61857.

[2] R. C. Young. *The algebra of many-valued quantities*. Mathematische Annalen, 1931. URL: https://doi.org/10.1007/BF01457934.

[3] J. Ide, E. Köbis, D. Kuroiwa, A. Schöbel, and C. Tammer. *The relationship between multi-objective robustness concepts and set-valued optimization*. Journal of Fixed Point Theory and Applications, 2014. URL: https://doi.org/10.1186/1687-1812-2014-83.

[4] R. S. Burachik, C. Y. Kaya, and M. M. Rizvi. *A New Scalarization Technique to Approximate Pareto Fronts of Problems with Disconnected Feasible Sets*. Journal of Optimization Theory and Applications, 2013. URL: https://doi.org/10.1007/s10957-013-0346-0.

[5] L. A. Soyster. *Convex Programming with Set-Inclusive Constraints and Application to Inexact Linear Programming*. Operations Research, Vol. 2, pp. 1154–1157, 1973.

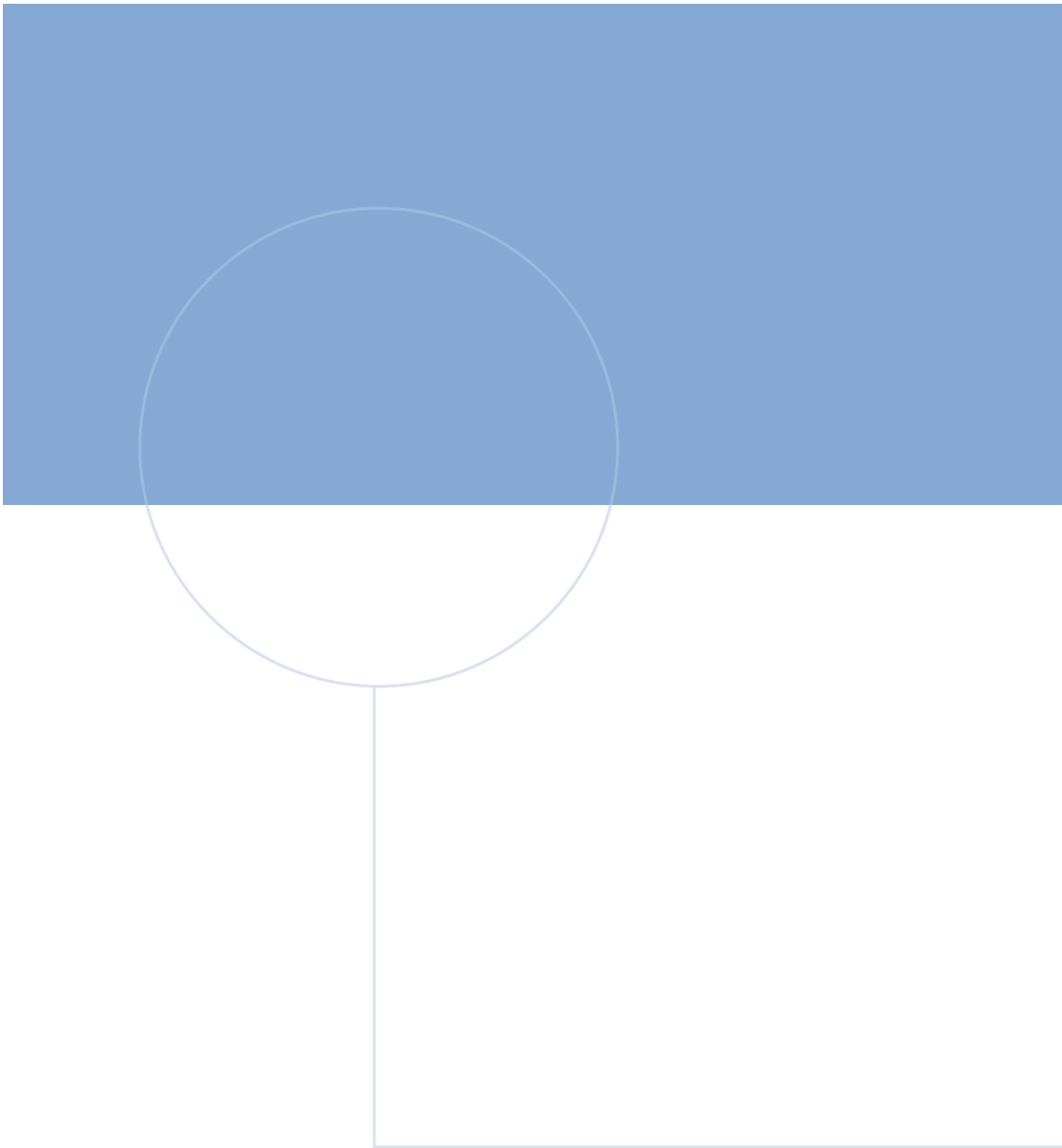
[6] A. H. Rønold. *Project thesis, TMA4500*. Department of Mathematical Sciences, Norwegian University of Science and Technology, 2020.

[7] M. Ehrgott. *Multicriteria Optimization*. Springer, Berlin, 2. Edition, 2005.

[8] A. S. M. Ehrgott J. Ide. *Minmax robustness for multi-objective optimization problems*. European Journal of Operational Research, 2014.

[9] J. Ide and E. Köbis. *Concepts of efficiency for uncertain multi-objective optimization problems based on set order relations*. Mathematical Methods of Operations Research, 2014. URL: https://doi.org/10.1007/s00186-014-0471-z.

[10] E. Köbis. *On robust optimization - a unified approach to robustness using a nonlinear scalarizing functional and relations to set optimization*. Universitäts- und Landesbibliothek Sachsen-Anhalt, 2014. URL: http://dx.doi.org/10.25673/1122.



NTNU

Norwegian University of
Science and Technology