August 2020

The Robustness Gap for Uncertain Multiobjective Linear Programs

Lena Fritzen

Clemson University, lena_f@gmx.de

The robustness gap for uncertain multiobjective linear programs

A Thesis
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
Mathematics

by
Lena Fritzen
August 2020

Accepted by:
Dr. Margaret Wiecek, Committee Chair
Dr. Matthew Saltzman
Dr. Yuyuan Ouyang

Abstract

In robust multiobjective optimization, a new robustness gap is defined in [4]. This gap measures the minimal distance between the robust Pareto set and the Pareto sets of all scenarios. Upper and lower bounds of this gap are derived for the convex case. In this thesis, a deeper examination into the definition and application of this gap for uncertain multiobjective linear programs is presented. Numerical examples are developed and results are reported for the first time.

Acknowledgments

I would like to take this opportunity to thank everyone who has supported me with my Master here in Clemson. A special thanks goes to my advisor, Dr. Wiecek, who has always taken the time to not only provide me with academic support, but has also helped me with my life here. Unfortunately, we are not able to say a proper goodbye. You were always available for my questions and concerns and showed me that is ok to ask for help. I would also like to thank the other members of my committee, Dr. Saltzman and Dr. Ouyang, for their time and helpful comments. Finally, I would like to thank all the professors, staff and students who have made my year here in Clemson a unique experience. I am grateful that Dr. Khan was always available for my questions on paperwork and I wish him the best for his next step in his career. Lauren and her family opened their home for me and made me feel welcome in America, Adobo will be part of my live now. Without all your help I would not have been able to get my degree during the hard times of COVID-19.

Table of Contents

List of Figures

Chapter 1

Introduction

In the business environment of companies, processes are often described as multiobjective programs with uncertainties. For example, managers have to make decisions about which products are ordered in which quantities and through which supply chains. These decisions are complicated by uncertainties such as future changes in demand and supply and the associated price fluctuations. When deciding which strategy to choose, it is often important for managers to weigh the right trade-off between security and the additional costs involved. The new robustness gap introduced in [4] can be a great help in this regard.

The goal of this thesis is to learn about uncertainty in multiobjective linear programming and the existence and properties of a robustness gap.

In Chapter 1, multiobjective linear programs are formulated and the concepts of robustness and uncertainty are introduced. The definition of a robustness gap is presented, and the gap is illustrated on a real life biobjective decision making situation under uncertainty.

In Chapter 2 and 3 two types of multiobjective linear programs are introduced. For each case, the gap is difficult to compute, so upper and lower bounds on this gap are developed. For the special case of biobjective linear programs with decision uncertainty we conduct numerical examples to further investigate the gap in Chapter 4. The thesis is concluded in Chapter 5 while the developed algorithms coded in MATLAB are contained in the Appendix.

1.1 Background

This thesis is based on [4], in which a general concept of robustness gap for multiobjective programs is introduced. Throughout this work, let $\mathbb{R}^l$ be the l-dimensional real vector space. Let $\|\cdot\|_p \colon \mathbb{R}^l \to \mathbb{R}$ denote a p-norm on $\mathbb{R}^l$ and $\|\cdot\|_q$ denote the dual (or polar) norm to $\|\cdot\|_p$ with the property that $\frac{1}{p} + \frac{1}{q} = 1$. Given these norms, we define the primal space $(\mathbb{R}^l, \|\cdot\|_p)$ and its dual space $(\mathbb{R}^l, \|\cdot\|_q)$. To keep notation short, we also refer to the primal space $(\mathbb{R}^l, \|\cdot\|_p)$ simply as $\mathbb{R}^l$. The cone defined by the nonnegative orthant of $\mathbb{R}^l$ is $\mathbb{R}^l_{\geqq} := \{y \in \mathbb{R}^l \mid y_i \geq 0 \ \forall i = 1, \ldots, l\}$ and we also refer to $\mathbb{R}^l_{\geq} := \mathbb{R}^l_{\geqq} \backslash \{0\}$ and $\mathbb{R}^p_{>} := \{y \in \mathbb{R}^l \mid y_i > 0 \ \forall i = 1, \ldots, l\}$. For all $y, z \in \mathbb{R}^l$, where $l \geq 2$, the order relations induced by $\mathbb{R}^l_{\geqq}$ are given by $y < z \Leftrightarrow y_i < z_i \ \forall 1 \leq i \leq l$; $y \leq z \Leftrightarrow y \neq z$ and $y_i \leq z_i \ \forall 1 \leq i \leq l$; and $y \leqq z \Leftrightarrow y_i \leq z_i \ \forall 1 \leq i \leq l$.

1.1.1 Multiobjective Linear Programs

Multiobjective linear programs occur in a lot of real-world applications where the conditions can be modeled with linear constraints, while different objectives can also be formulated as linear functions.

Definition 1. We consider the *multiobjective linear program* MOLP

$$\left(\begin{array}{lrrcl} \text{(MOLP)} & \min & Cx & & \\ & \text{s. t.} & Ax & \geqq & b \\ & & x & \in & \mathbb{R}^n \end{array}\right), \tag{1.1}$$

where $C \in \mathbb{R}^{l\times n}$, $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$.

In the following, A_i describes the i-th row of A and respectively C_i the i-th row of C.

In contrast to linear programs (LPs), optimization in (1.1) is not performed according to a single scalar-valued function, but several objective functions simultaneously, which do not have to assume their minima at the same feasible point. Thus, there is not one optimal solution, but a set of Pareto optimal solutions.

Definition 2. A vector $y^* \in \mathcal{Y} = C(X) := \{Cx|x \in X\}$ is called a *Pareto objective vector* of MOLP if there is no $y \in \mathcal{Y}$ with the property $y \leq y^*$. A feasible solution $x^* \in X$ is called an *efficient solution* for MOLP if the image Cx^* is a Pareto objective vector of $\mathcal{Y}$. The *efficient set* of MOLP is denoted as X_E and it is also referred to as the *solution set* or the *set of efficient decisions.* The *Pareto set* of MOLP is denoted as $\mathcal{Y}_P$ and it is also referred to as the set of *Pareto criterion vectors* or *Pareto outcomes* that represent the performance of the efficient decisions with respect to the objective functions.

Another possibility to look at MOLPs is to apply the weighted-sum scalarization. This means that every objective target gets a specific weight λ_i for $i = 1, \ldots, l$ where λ sums up to 1 in the dual norm. Applying this concept, we receive an LP which can be solved with well-known algorithms.

Definition 3. The *weighted-sum scalarization* of MOLPs is defined as

$$\left(\begin{array}{lllll} (LP(\lambda)) & \min & \lambda^T Cx & & \\ & \text{s. t.} & Ax & \geqq & b \\ & & x & \in & \mathbb{R}^n \end{array} \right) \tag{1.2}$$

In fact, all optimal solutions of the weighted-sum scalarization (1.2) where $\lambda \in \mathbb{R}^l_{\geq}$ are efficient solutions for (1.1), see [2].

Theorem 1. Let $\bar{x}$ be an optimal solution of $LP(\lambda)$ of the form (1.2) where $\lambda \in \mathbb{R}^l_{\geq}$, then $\bar{x}$ is efficient for MOLP of the form (1.1).

1.1.2 Uncertainty and Robustness

In real-life, uncertainty occurs all the time when information is unknown. Decisions today require us to think about the future, which can be modeled as a collection of scenarios. However, only one scenario in this collection will actually happen. Nevertheless, we consider all possible scenarios when making a decision today and therefore we include uncertainty in the optimization problem.

Definition 4. We consider the *uncertain multiobjective linear optimization program* $\{\text{MOLP}(\xi)\}_{\xi \in \mathcal{U}}$

$$\left\{ \begin{array}{lll} (\text{MOLP}(\xi)) & \min\limits_{x} & Cx \\ & \text{s. t.} & A(x+\eta) \geqq b \\ & & x \in \mathbb{R}^n \end{array} \right\}_{\xi=(\eta,C,A,b)\in\mathcal{U}} , \tag{1.3}$$

where $\mathcal{U} \subset \mathbb{R}^n \times \mathbb{R}^{l\times n} \times \mathbb{R}^{m\times n} \times \mathbb{R}^m$ is the *uncertainty set.* The elements $\xi \in \mathcal{U}$ are called *scenarios* of $\mathcal{U}$. In the case of MOLPs, decision uncertainty can be contained in the vector of decision variables $x \in \mathbb{R}^n$, described via η, and parameter uncertainty can be contained in the problem data $(C, A, b) \in \mathbb{R}^{l \times n} \times \mathbb{R}^{m \times n} \times \mathbb{R}^m$.

For a fixed scenario $\xi \in \mathcal{U}$, it is assumed that no uncertainty is present in MOLP(ξ), i.e., all data of MOLP(ξ) is known. Therefore, for each $\xi \in \mathcal{U}$, MOLP(ξ) is considered a deterministic MOLP.

For this thesis, we assume that $\mathcal{U}$ is a **compact** and **convex** set.

Definition 5. For every scenario $\xi = (\eta, C, A, b) \in \mathcal{U}$, the *feasible set* of MOLP(ξ) is denoted as

$$X(\xi) := \{x \in \mathbb{R}^n \mid A(x + \eta) \geqq b\}. \tag{1.4}$$

For every $\xi \in \mathcal{U}$, the *outcome set* of MOLP(ξ), which is the image set of the feasible set $X(\xi)$, is given as

$$\mathcal{Y}(\xi) := \{y \in \mathbb{R}^l \mid \exists x \in X(\xi) \colon y = Cx\} = C(X(\xi)). \tag{1.5}$$

The set $\mathcal{Y}(\xi)$ is compact and convex since $X(\xi)$ is assumed to be compact and convex and C is a linear operator and therefore the function $C(\cdot)$ is continuous.

For defining the feasibility of the uncertain multiobjective linear problem $\{\text{MOLP}(\xi)\}_{\xi \in \mathcal{U}}$, we define a concept of robustness.

Definition 6. A point $x^* \in \mathbb{R}^n$ is called a *robust feasible* solution to $\{\text{MOLP}(\xi)\}_{\xi \in \mathcal{U}}$ if x^* is feasible for all possible realizations of uncertainty, i.e., if

$$x^* \in X^{\text{RC}} := \{x \in \mathbb{R}^n \mid A(x + \eta) \geqq b \;\forall \xi = (\eta, C, A, b) \in \mathcal{U}\}. \tag{1.6}$$

The set X^{RC} is called the *robust feasible set*. The *robust outcome set*, which is the image of the robust feasible set X^{RC}, is denoted as

$$\mathcal{Y}^{\mathrm{RC}} := \{C^{RC}(x) \mid x \in X^{\mathrm{RC}}\} = C^{RC}(X^{\mathrm{RC}}), \tag{1.7}$$

where $C^{RC} \colon \mathbb{R}^n \to \mathbb{R}^l$ is given as

$$C_i^{RC}(x) = \sup_{\xi \in \mathcal{U}} C_i x$$

for all $1 \leq i \leq l$ and for any $x \in \mathbb{R}^n$.

As a consequence of (1.6) and (1.7), we obtain

$$X^{\mathrm{RC}} = \bigcap_{\xi \in \mathcal{U}} X(\xi), \quad \mathcal{Y}^{\mathrm{RC}} \subseteq \bigcap_{\xi \in \mathcal{U}} \mathcal{Y}(\xi) + \mathbb{R}^p_{\geqq}.$$

Hence, the robust feasible set X^{RC} is compact and convex as an intersection of compact and convex sets. Because C is a linear operator and therefore the function $C(\cdot)$ is continuous, the robust outcome set $\mathcal{Y}^{\mathrm{RC}}$ is also compact and convex. The robust Pareto set, Y_P^{RC}, can be interpreted as the set of conservative decisions but making the user act safely in every scenario $\xi \in \mathcal{U}$.

1.2 Robustness Gap

In the literature, the robustness gap is initially defined for uncertain single-objective optimization problems as a measure of the distance between the robust optimal objective value and the optimal objective values of the scenarios [1].

Definition 7. We consider the *uncertain single-objective linear optimization program* formulated as

$$\left\{\begin{array}{lll} (LP^{\text{s-o}}(\xi)) & \min\limits_{x} & cx \\ & \text{s. t.} & A(x+\eta) \geqq b \\ & & x \in \mathbb{R}^n \end{array}\right\}_{\xi=(\eta,c,A,b)\in\mathcal{U}}, \tag{1.8}$$

where $\mathcal{U} \subset \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^{m\times n} \times \mathbb{R}^m$.

The *single-objective robustness gap* is defined as

$$\vartheta^{\text{s-o}} := \min_{x\in X^{\text{RC}}} c^{\text{RC}}(x) - \sup_{\xi\in\mathcal{U}} \min_{x\in X(\xi)} cx. \tag{1.9}$$

In [4], the authors intend to formulate a similar concept for uncertain multiobjective problems and therefore define a multiobjective robustness gap. This gap represents the smallest distance between the sets $\mathcal{Y}_P^{\text{RC}}$ and $\bigcup_{\xi\in\mathcal{U}} \mathcal{Y}_P(\xi)$ that is measured with a chosen norm.

Definition 8. For each $\xi \in \mathcal{U}$ let

$$\vartheta(\xi) := \operatorname{dist}\left(\mathcal{Y}_P^{\text{RC}}, \mathcal{Y}_P(\xi)\right) = \inf_{z\in\mathcal{Y}_P^{\text{RC}}} \inf_{y\in\mathcal{Y}_P(\xi)} \|z-y\|$$

and define the *multiobjective robustness gap* as

$$\vartheta := \inf_{\xi\in\mathcal{U}} \vartheta(\xi) = \inf_{\xi\in\mathcal{U}} \operatorname{dist}\left(\mathcal{Y}_P^{\text{RC}}, \mathcal{Y}_P(\xi)\right).$$

Because this thesis only deals with p-norms and convex and compact uncertainty sets $\mathcal{U}$, this definition can be reformulated as

$$\vartheta = \min_{\xi\in\mathcal{U}} \inf_{z\in\mathcal{Y}_P^{\text{RC}}} \inf_{y\in\mathcal{Y}_P(\xi)} \|z-y\|_p. \tag{1.10}$$

It is also possible to define the gap using the weighted-sum scalarization 1.2 rather than the explicit definition of the Pareto sets:

$$\vartheta = \min_{\xi \in \mathcal{U}} \inf_{\substack{\lambda_1, \lambda_2 \geqq 0: \\ \|\lambda_1\|_q = \|\lambda_2\|_q = 1}} \inf_{z \in \arg\min_{\bar{z} \in \mathcal{Y}^{RC}} \lambda_1^T \bar{z}} \inf_{y \in \arg\min_{\bar{y} \in \mathcal{Y}(\xi)} \lambda_2^T \bar{y}} \|z - y\|_p.$$

Note, that for the single-objective case the definition of ϑ leads to the same result as $\vartheta^{\text{s-o}}$. For the single-objective case, $\mathcal{Y}_P^{RC}, \mathcal{Y}_P(\xi) \subset \mathbb{R}$ are singletons. So we can rewrite them as $\mathcal{Y}_P^{RC} = \{\bar{z}\}$ and $\mathcal{Y}_P(\xi) = \{\bar{y}\}$ with the relation that $\bar{z} \geq \bar{y}$ for all scenarios ξ. Because of this relation, the gap can be rewritten as

$$\vartheta = \inf_{z \in \mathcal{Y}_P^{\text{RC}}} z - \max_{\xi \in \mathcal{U}} \sup_{y \in \mathcal{Y}_P(\xi)} y$$

$$= \min_{z \in \mathcal{Y}_P^{\text{RC}}} z - \max_{\xi \in \mathcal{U}} \min_{y \in \mathcal{Y}_P(\xi)} y$$

$$= \min_{x \in X^{RC}} c^{RC}(x) - \max_{\xi \in \mathcal{U}} \min_{x \in X(\xi)} cx.$$

It turns out, that even for the linear case, finding the robustness gap is a difficult task, because $\mathcal{Y}_P^{\text{RC}}$ and $\mathcal{Y}_P(\xi)$ are not convex in general. Because of this, upper and lower bounds for the robustness gap are proposed.

1.2.1 Bounds on the Robustness Gap

We define the bounds as follows:

Definition 9. For each $\xi \in \mathcal{U}$ and for each $\lambda \in \mathbb{R}^p_{\geqq}$, we define

$$\Delta(\xi, \lambda) := \min_{z \in \mathcal{Y}^{\text{RC}}} \lambda^T z - \min_{y \in \mathcal{Y}(\xi)} \lambda^T y. \tag{1.11}$$

We can reformulate this as

$$\Delta(\xi,\lambda) = \min_{z\in\mathcal{Y}_P^{\mathrm{RC}}} \max_{y\in\mathcal{Y}_P(\xi)} \lambda^T(z-y).$$

Furthermore, we define the *lower robustness bound* and *upper robustness bound* as

$$\Delta^L := \inf_{\xi\in\mathcal{U}} \min_{\substack{\lambda\in\mathbb{R}^p_{\geq},\\ \|\lambda\|_q=1}} \Delta(\xi,\lambda) \qquad \text{and} \qquad \Delta^U := \inf_{\xi\in\mathcal{U}} \max_{\substack{\lambda\in\mathbb{R}^p_{\geq},\\ \|\lambda\|_q=1}} \Delta(\xi,\lambda). \tag{1.12}$$

The bounds on the robustness gap are given as the value of the weighted sums of the smallest distance between the Pareto set of the robust counterpart $\mathcal{Y}_P^{RC}$ and the union of all Pareto sets of the scenarios $\bigcup_{\xi\in\mathcal{U}} Y_P(\xi)$ measured with weight $\lambda\in\mathbb{R}^n_{\geq}$. For the lower bound the weight λ is chosen such that this value becomes minimal, while for the upper bound the weight λ is chosen such that this value becomes maximal. While the robustness gap is difficult to compute, because we look at the non-convex Pareto sets, these bounds use the weighted-sum minimum of $\mathcal{Y}^{\mathrm{RC}}$ and $\mathcal{Y}(\xi)$, which are compact and convex sets in the case of MOLPs.
In [4] it is shown that for all MOLPs with the distance measured with a p-norm, these bounds exists and fulfill

$$0 \leq \Delta^L \leq \vartheta \leq \Delta^U.$$

For the single-objective case, upper and lower bounds are tight, so $0 \leq \Delta^L = \vartheta^{\text{s-o}} = \Delta^U$.

1.3 Application

In this section, we illustrate the concept of the robustness gap on a real-life example and develop a numerical example. Assume a company produces two types of face masks during COVID-19. Every evening the manager has to decide how much material to order for the next day production. He has the option to order two types of material from two different suppliers which are in a competition with each other. His goals are to minimize the cost of the ordered material from both suppliers respectively subject to production requirements and workers availability. Because the suppliers are in competition with each other and the manager must maintain good relations with each of them, we consider the costs of the two different materials separately. At the same time, he intends to satisfy the demand for face masks from different hospitals and companies for next day. He also has to consider that his company can only process a certain amount of material every day.

Another important thing is that while he has to order a certain amount of the different materials for next day, he does not know how expensive this material might be, how many masks will be ordered next morning or how many people will be allowed to work in the company next morning, because all these factors depend on the number of new COVID-19 cases on the actual day, which will only be announced late at night. He only knows that there are two possible scenarios depending on whether the actual number of COVID-19 cases in his area is below or above a certain number.

This problem can be modeled as an MOLP with uncertainty in C and b. By plugging in certain numbers for the needed material per mask, the number of masks for the scenarios and the price of new material for the scenarios, we get a system where x_1 is the number of rolls of material from the first supplier ordered for the unknown price ξ_1. Similar, x_2 is the number of rolls of material from the second supplier ordered for the unknown price ξ_2. The also unknown numbers of orders of masks in hundreds of the first and second type are described as ξ_3 and respectively ξ_4. We know that we can produce 300 masks of type one and 100 mask of type two with one roll of material from the first supplier. Similar, we can produce 100 mask of type one and 400 masks of type two from one roll of material from the second supplier. So the total number of produced masks in hundreds of type one is equal to three times the number of rolls from the first supplier ($3x_1$) plus the number of rolls from the second supplier (x_2), while the number of produced masks in hundreds of type two is equal to the number of rolls from the first supplier (x_1) plus four times the number of rolls from the second supplier ($4x_2$).

Based on the unknown number of workers for next day, the number of masks in hundreds we are able to produce is ξ_5, while the total number of produced masks is four hundred masks out of every roll of material from the first supplier plus five hundred masks out of every roll of material from the second supplier. We have to find the number of rolls we should order from every supplier which minimize the price we have to pay to every supplier respectively.

The manager has to consider two scenarios depending on the number of COVID-19 cases. If the number of cases is above a certain number, scenario ξ^1 with $\xi_1^1, \ldots, \xi_5^1$ will happen, else scenario ξ^2 with $\xi_1^2, \ldots, \xi_5^2$ will happen.

Written as an uncertain MOLP we get

$$\left\{ \begin{array}{lllll} (MOLP(\xi)) & \min\limits_x & \begin{pmatrix} \xi_1 & 0 \\ 0 & \xi_2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} & & \\ & \text{s. t.} & 3x_1 + x_2 & \geq \xi_3 & \\ & & x_1 + 4x_2 & \geq \xi_4 & \\ & & 4x_1 + 5x_2 & \leq \xi_5 & \\ & & x_1, x_2 & \geq 0 & \end{array} \right\}_{\xi \in \mathcal{U}},$$

where $\mathcal{U} = \left\{ \xi^1 := (2, 3, 14, 12, 48)^T, \xi^2 := (4, 1, 11, 22, 55)^T \right\} \subseteq \mathbb{R}^5$.

This uncertain MOLP can be graphically represented in the following figure.

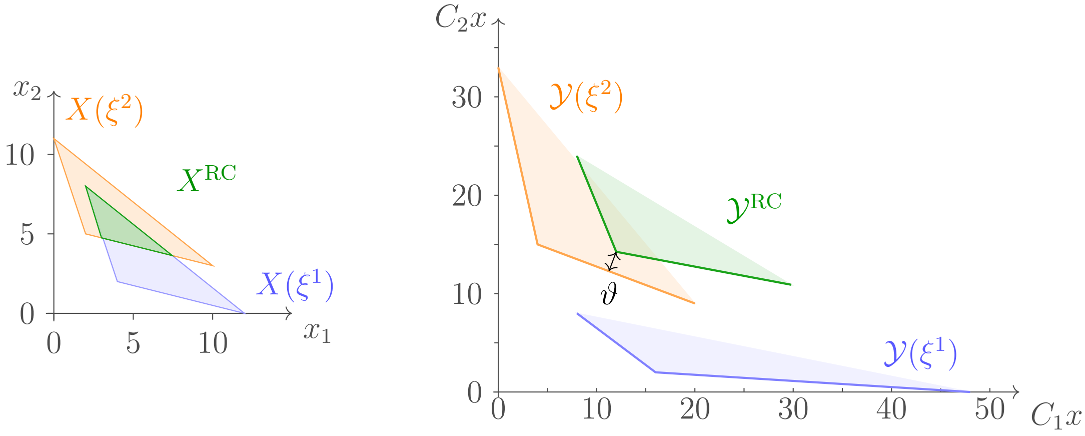


Figure 1.1: Left: the feasible sets. Right: the outcome sets; the thick solid lines depict the Pareto sets $\mathcal{Y}_P(\xi^1)$, $\mathcal{Y}_P(\xi^2)$, and $\mathcal{Y}_P^{\text{RC}}$ and ϑ illustrates the robustness gap.

Now the manager has to decide how much material to order. He has to decide if he wants to choose a solution which works well for both possible scenarios, a robust efficient solution $x \in X_E^{RC}$, or based on his knowledge of the case numbers of the previous days, of he wants to take risk and assume that one scenario is more likely and choose to order an amount which works well for this special scenario $x \in X_E(\xi_i)$ $i \in \{1, 2\}$. He also has to decide which specific amount to order, because all these efficient sets include infinitely many points.

The robustness gap ϑ with the Euclidean norm can support manager's decision making. Based on this gap, he can see his minimum loss when he decides to choose a robust efficient solution instead of a solution which is in the efficient set of the actual scenario. He would also see which point of the robust Pareto set and which scenario with which Pareto point in this scenario lead to the gap and use this information to decide how much material to order.

In the case of the MOLP above, there is no easy way to compute the gap. So we find the gap by inspection:

In Figure 1.1 we can see that the shortest distance between the Pareto set $\mathcal{Y}_P^{RC}$ and the Pareto sets $\mathcal{Y}_P(\xi_1)$ and $\mathcal{Y}_P(\xi_2)$ is between the point $\bar{z} = (12, 14.25)^T \in \mathcal{Y}_P^{\mathrm{RC}}$ and the line segment from (4,15) to (20,9) in $\mathcal{Y}_P(\xi^2)$. By a simple computation, we get

$$\vartheta = \min_{\xi \in \mathcal{U}} \min_{z \in \mathcal{Y}_P^{\mathrm{RC}}} \min_{y \in \mathcal{Y}_P(\xi)} \|z - y\|_2 = \|\bar{z} - \bar{y}\|_2 = \frac{18}{\sqrt{73}} \approx 2.11,$$

where $\bar{y} = (\frac{822}{73}, \frac{3585}{292})^T \in \mathcal{Y}_P(\xi^2)$. These vectors $\bar{z}$ and $\bar{y}$ are the images of the efficient vectors $x^{RC} = (3, 19/4)^T \in X_E^{RC}$ and $x(\xi) = (\frac{411}{146}, \frac{3585}{292})^T \in X_E(\xi^2)$. But this gap is only one specific number resulting from the distance measured in the Euclidean norm. Maybe the manager wants to get more information, especially because he does not think that that the objective functions have the same importance or if he wants to consider more suppliers, so he gets more objective functions.

Mathematically it is already difficult to solve for the gap when considering two objective functions. Considering even more could be impossible to solve, so we have to use the concepts of the upper and lower bound which provide more information with selected weights for the weighted-sum scalarization and give an interval where the true robustness gap is in between.

In order to calculate Δ^L, we consider $\tilde{z} := (8, 24)^T \in \mathcal{Y}^{\mathrm{RC}}$, $\tilde{y} := (8, 8)^T \in \mathcal{Y}(\xi^1)$ and $\tilde{\lambda} := (1, 0)^T \in \mathbb{R}^2_{\geq}$, and since in this example the primal and the dual space are both $(\mathbb{R}^2, \|\cdot\|_2)$, we note $\|\lambda\|_2 = 1$. Then applying (1.11) and (1.12) we obtain

$$0 \leq \Delta^L = \inf_{\xi \in \mathcal{U}} \min_{\substack{\lambda \in \mathbb{R}^2_{\geq}, \\ \|\lambda\|_2 = 1}} \left(\min_{z \in \mathcal{Y}^{\mathrm{RC}}} \lambda^T z - \min_{y \in \mathcal{Y}(\xi)} \lambda^T y \right) \leq \Delta(\xi^1, \tilde{\lambda}) \leq \tilde{\lambda}^T \tilde{z} - \tilde{\lambda}^T \tilde{y} = 8 - 8 = 0,$$

where the first inequality results from the fact that the minimal weigthed-sum value of the robust counterpart is always at least as big as the minimal weighted-sum value of a scenario. Consequently, we obtain $\Delta^L = 0$ as displayed in Figure 1.2 below, in which the longer vertical dotted line represents the two level curves of value $\tilde{\lambda}^T \tilde{z} = \tilde{\lambda}^T \tilde{y} = 8$.

In order to determine Δ^U, we first calculate $\max_{\substack{\lambda \in \mathbb{R}^2_{\geq}, \\ \|\lambda\|_2 = 1}} \Delta(\xi^2, \lambda)$. For any $\lambda \in \mathbb{R}^2_{\geq}$, both sets $\arg\min_{z \in \mathcal{Y}^{\mathrm{RC}}} \lambda^T z$ and $\arg\min_{y \in \mathcal{Y}(\xi^2)} \lambda^T y$ contain at least one extreme point (EP) in $\mathcal{Y}_P^{\mathrm{RC}}$ and $\mathcal{Y}_P(\xi^2)$, respectively. Given the coordinates of the EPs of the latter two sets, for each of these sets we find the intervals for $\lambda_1 \geq 0$ such that for all λ_1 in each interval the same EP is an optimal solution to the associated weighted-sum problem. We then merge the intervals and establish new intervals of λ_1 such that for every λ_1 in each new interval the same points $\bar{z} \in \mathcal{Y}_P^{\mathrm{RC}}$ and $\bar{y} \in \mathcal{Y}_P(\xi^2)$ are optimal solutions to the associated weighted-sum problems. Hence, we partition $\Lambda := \{\lambda \in \mathbb{R}^2_{\geq} \mid \|\lambda\|_2 = 1\}$ into five subsets Λ_i such that $\Lambda = \bigcup_{i=1}^5 \Lambda_i$, where
$\Lambda_1 := \{\lambda \in \Lambda \mid \lambda_1 \in [0, 0.185]\}$, $\Lambda_2 := \{\lambda \in \Lambda \mid \lambda_1 \in [0.185, 0.351]\}$,
$\Lambda_3 := \{\lambda \in \Lambda \mid \lambda_1 \in [0.351, 0.925]\}$, $\Lambda_4 := \{\lambda \in \Lambda \mid \lambda_1 \in [0.925, 0.976]\}$ and
$\Lambda_5 := \{\lambda \in \Lambda \mid \lambda_1 \in [0.976, 1.0]\}$.
We then solve $\max_{\lambda \in \Lambda_i} \Delta(\xi^2, \lambda)$ for each $i = 1, \ldots, 5$ and obtain

$$\max_{\lambda \in \Lambda} \Delta(\xi^2, \lambda) = \max_{i=1,\ldots,5} \max_{\lambda \in \Lambda_i} \Delta(\xi^2, \lambda) = \max\{3.681, 3.681, 7.115, 7.120, 8\} = 8$$

for $\bar{\lambda} = (1,0)^T, \bar{z} = (8,24)^T, \bar{y} = (0,33)^T$. Next, we show

$$\max_{\substack{\lambda \in \mathbb{R}^2_{\geq}, \\ \|\lambda\|_2 = 1}} \Delta(\xi^2, \lambda) < \max_{\substack{\lambda \in \mathbb{R}^2_{\geq}, \\ \|\lambda\|_2 = 1}} \Delta(\xi^1, \lambda).$$

For $\lambda' := (0,1)^T$ we obtain

$$\max_{\substack{\lambda \in \mathbb{R}^2_{\geq}, \\ \|\lambda\|_2 = 1}} \Delta(\xi^1, \lambda) \geq \Delta(\xi^1, \lambda') = 12 - 0 = 12 > 8 = \max_{\substack{\lambda \in \mathbb{R}^2_{\geq}, \\ \|\lambda\|_2 = 1}} \Delta(\xi^2, \lambda).$$

Therefore,

$$\Delta^U = \inf_{\xi \in \mathcal{U}} \max_{\substack{\lambda \in \mathbb{R}^2_{\geq}, \\ \|\lambda\|_2 = 1}} \Delta(\xi, \lambda) = \max_{\substack{\lambda \in \mathbb{R}^2_{\geq}, \\ \|\lambda\|_2 = 1}} \Delta(\xi^2, \lambda) = 8.$$

The upper robustness bound Δ^U is also displayed in Figure 1.2 below.

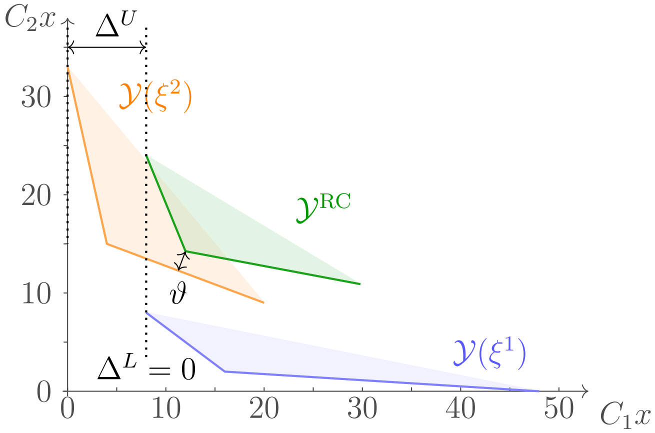


Figure 1.2: The lower bound Δ^L, the upper bound Δ^U, and the robustness gap ϑ.

The upper and lower bounds do not only give an interval within which the real robustness gap is located, but also include more information. While solving for the bounds, we also compute optimal vectors $\bar{\xi}$, $\bar{\lambda}$, $\bar{y}$ and $\bar{z}$ which lead to these bounds. The bounds are computed using scalarized weighted-sums based on $\bar{\lambda}$. If the manager thinks that the different goals have different priorities, he could use those values as a reference, especially if one of the obtained $\bar{\lambda}$ of the scalarized weighted-sum problem reflects his preferences. He could also compute the bounds for a special scenario ξ which would give him $\Delta^L(\xi)$ and $\Delta^U(\xi)$, for the multiobjective scenario robustness gap $\vartheta(\xi)$ with the properties: $\Delta^L \leq \Delta^L(\xi) \leq \vartheta(\xi) \leq \Delta^U(\xi)$ and $\Delta^U \leq \Delta^U(\xi)$.
All in all, the gap and the bounds can support the manager making a decision but do not provide sufficient information to replace him.

Chapter 2

Decision Uncertainty

In this section, we focus on the MOLPs with an uncertainty in the decision variable x. So the general uncertainty vector $\xi = (\eta, C, A, b) \in \mathcal{U}$ is reduced to $\xi = \eta \in \mathcal{U}$, because C, A and b are fixed. In applications this type of uncertainty occurs when it cannot be guaranteed that theoretical results can be implemented one-to-one based on implementation errors. We assume that we can model the decision uncertainty as we can add an uncertainty $\xi \in \mathcal{U}$ to the decision vector x. As stated in Chapter 1, we assume that $\mathcal{U}$ is compact and convex, and in some parts of this chapter we discuss the cases where $\mathcal{U}$ is a polytope.

Decision uncertainty leads to the following uncertain MOLP:

$$\left\{ \begin{array}{llrcl} (\mathrm{MOLP}(\xi)) & \min & Cx & & \\ & \text{s. t.} & A(x+\xi) & \geqq & b \\ & & x & \in & \mathbb{R}^n \end{array} \right\}_{\xi \in \mathcal{U}}, \tag{2.1}$$

where $\mathcal{U} \subset \mathbb{R}^n$.

For this class of MOLPs

$$X(\xi) = \{x \in \mathbb{R}^n | A(x + \xi) \geqq b\} \tag{2.2}$$

and we can define $X(0)$ as

$$X(0) := \{x \in \mathbb{R}^n | Ax \geqq b\}.$$

Note, that $X(0)$ does not have to be a subset of the union of all $X(\xi)$.

It turns out that for MOLPs with decision uncertainty, the union of the feasible sets have a special form.

Proposition 1. The union of all feasible sets in (2.2) can be represented as a Minkowski difference

$$\bigcup_{\xi \in \mathcal{U}} X(\xi) = X(0) \ominus \mathcal{U} := \{x - \xi | x \in X(0),\ \xi \in \mathcal{U}\}.$$

Proof. "$\subseteq$" Assume $x \in \cup_{\xi \in \mathcal{U}} X(\xi)$, then there exists $\xi_x \in \mathcal{U}$ so that $x \in X(\xi_x)$. Define $\bar{x} := x + \xi_x$, then $x = \bar{x} - \xi_x$ with $\xi_x \in \mathcal{U}$ and $A\bar{x} = A(x + \xi_x) \geqq b$, because $x \in X(\xi_x)$. So $\bar{x} \in X(0)$ and therefore $x \in X(0) \ominus \mathcal{U}$.
"$\supseteq$" Assume $\bar{x} = x - \bar{\xi} \in X(0) \ominus \mathcal{U}$. Then $\bar{\xi} \in \mathcal{U}$ and $A(\bar{x} + \bar{\xi}) = A(x - \bar{\xi} + \bar{\xi}) = Ax \geqq b$, because $x \in X(0)$. So $\bar{x} \in X(\bar{\xi})$ and therefore $\bar{x} \in \cup_{\xi \in \mathcal{U}} X(\xi)$. □

Corollary 1. The union of all feasible sets $\bigcup\limits_{\xi \in \mathcal{U}} X(\xi)$ is a convex set.

Proof. Because $X(0)$ as a polyhedron and $\mathcal{U}$ by definition are convex and the Minkowski difference of two convex sets is convex, the union of all feasible sets is also convex. □

To understand (2.1) better, we analyze the associated robust counterpart and present formulations of the optimization problems whose optimal solutions determine the upper and lower bound of the robustness gap.

In this chapter, we illustrate the general concepts of Chapter 1 in the special case of decision uncertainty on a simple example. In Chapter 4 we present more biobjective examples with decision uncertainty and compute the robustness gaps and lower and upper bounds.

Example 2.0.1. We consider the 2-dimensional decision uncertainty MOLP with

$$C = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \; A = \begin{pmatrix} -\frac{1}{6} & 1 \\ 7 & -1 \\ 1 & 1 \end{pmatrix}, \; b = \begin{pmatrix} -\frac{1}{6} \\ 3 \\ 1 \end{pmatrix}, \; \mathcal{U} = \left\{ \xi \in \mathbb{R}^2 \mid \xi_1, \xi_2 \geqq 0, \; \xi_1 + \xi_2 = 1 \right\}.$$

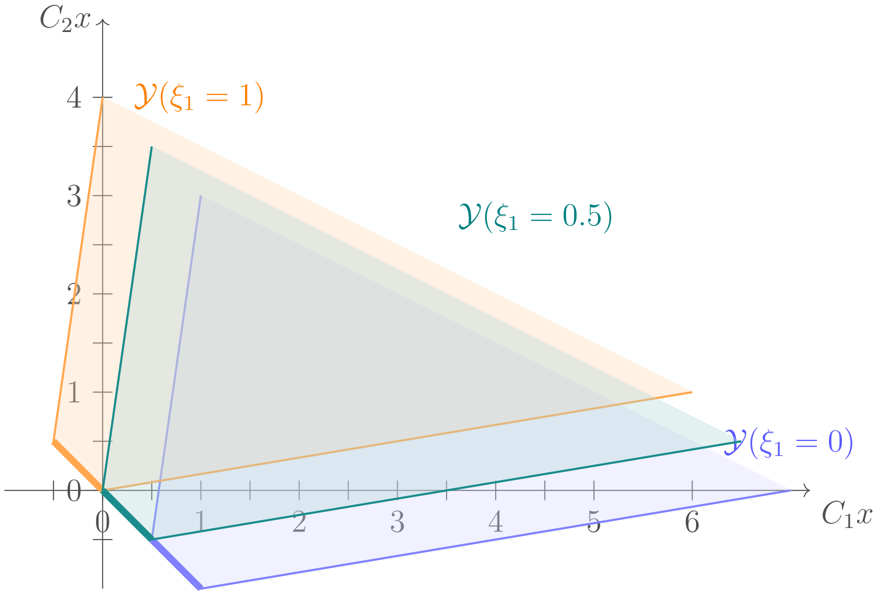


Figure 2.1: Outcome sets of Example 2.0.1 for specific values of the uncertainty parameter ξ.

Because $C = I$, we know that $C_1 x = x_1$ and $C_2 x = x_2$, so $X(\xi) = \mathcal{Y}(\xi)$. Based on C, we want to minimize either x_1 or x_2, so the union of all Pareto sets $\mathcal{Y}_P(\xi)$ for of all uncertainties in $\mathcal{U}$ build the line segment between $(1, -1)$ and $(-0.5, 0.5)$.

2.1 Robust Counterpart

To find the robust counterpart to the family of MOLPs of form (2.1), we identify the robust feasible set defined as the set of decision vectors which are feasible for all possible uncertainty vectors in $\mathcal{U}$. Given

$$X^{RC} := \{x \in \mathbb{R}^n \mid A(x + \xi) \geqq b, \ \forall \xi \in \mathcal{U}\},$$

we can rewrite this as

$$X^{RC} = \{x \in \mathbb{R}^n \mid Ax \geqq b - A\xi, \ \forall \xi \in \mathcal{U}\}.$$

Because $\mathcal{U}$ is a compact set, every component in the right hand side of the inequality constraint above takes its minimum for a $\xi \in \mathcal{U}$. Defining $\tilde{b} := b - (\min_{\xi \in \mathcal{U}} A_1 \xi \ldots, \min_{\xi \in \mathcal{U}} A_m \xi)^T$, it is easy to see that

$$X^{RC} = \left\{x \in \mathbb{R}^n \mid Ax \geqq \tilde{b}\right\}, \tag{2.3}$$

and we can compute the robust counterpart of the decision uncertainty problem after finding $\tilde{b}$. The difficulty to find $\tilde{b}$ depends on the structure of $\mathcal{U}$. If $\mathcal{U}$ is a polytope given as $\mathcal{U} = \{\xi \in \mathbb{R}^n \mid E\xi \geqq e\}$, we can find $\tilde{b}$ by solving the linear programs of the form

$$\left(\begin{array}{llllll} (LP(\xi)) & \tilde{b}_i = b_i & - & \min & A_i\xi & \\ & & & \text{s. t.} & E\xi & \geqq & e \end{array}\right)$$

for $i = 1, \ldots, m$. With this finite definition (2.3), we can formulate the robust counterpart as a deterministic MOLP with finite constraints:

$$\left(\begin{array}{lllll} \left(RC^{MOLP_{Decision}}\right) & \min & Cx & & \\ & \text{s. t.} & Ax & \geqq & \tilde{b} \\ & & x & \in & \mathbb{R}^n \end{array}\right).$$

Example 2.1.1. Rewriting Example 2.0.1, we get the collection of MOLPs

$$\left\{\begin{array}{lllll} (MOLP(\xi)) & \min\limits_{x} & (x_1 + \xi_1, x_2 + \xi_2)^T & & \\ & \text{s. t.} & -\frac{1}{6}x_1 + x_2 & \geq & -\frac{1}{6} + -\frac{1}{6}\xi_1 - \xi_2 \\ & & 7x_1 - x_2 & \geq & 3 - 7\xi_1 + \xi_2 \\ & & x_1 + x_2 & \geq & 1 - \xi_1 - \xi_2 \end{array}\right\}_{\xi \in \mathcal{U}},$$

where $\mathcal{U} = \{\xi \in \mathbb{R}^2 \mid \xi_1, \xi_2 \geqq 0,\ \xi_1 + \xi_2 = 1\}$.

We can find $\tilde{b}_1$ by solving the linear program

$$\left(\begin{array}{llllrcl} (LP(\xi)) & \tilde{b}_1 = & -\frac{1}{6}- & \min & -\frac{1}{6}\xi_1 + \xi_2 & & \\ & & & \text{s. t.} & \xi_1 & \geqq & 0 \\ & & & & \xi_2 & \geqq & 0 \\ & & & & \xi_1 + \xi_2 & \geqq & 1 \\ & & & & -\xi_1 - \xi_2 & \geqq & -1 \end{array} \right).$$

This linear program takes its minimum for $\xi_1 = 1$ and $\xi_2 = 0$, so $\tilde{b}_1 = -\frac{1}{6} - -\frac{1}{6} = 0$. Similar, we get $\tilde{b}_2 = 4$ and $\tilde{b}_3 = 0$ and rewrite the robust feasible set as

$$X^{RC} = \left\{ x \in \mathbb{R}^3 \mid \begin{pmatrix} -\frac{1}{6} & 1 \\ 7 & -1 \\ 1 & 1 \end{pmatrix} x \geqq \begin{pmatrix} 0 \\ 4 \\ 0 \end{pmatrix} \right\}.$$

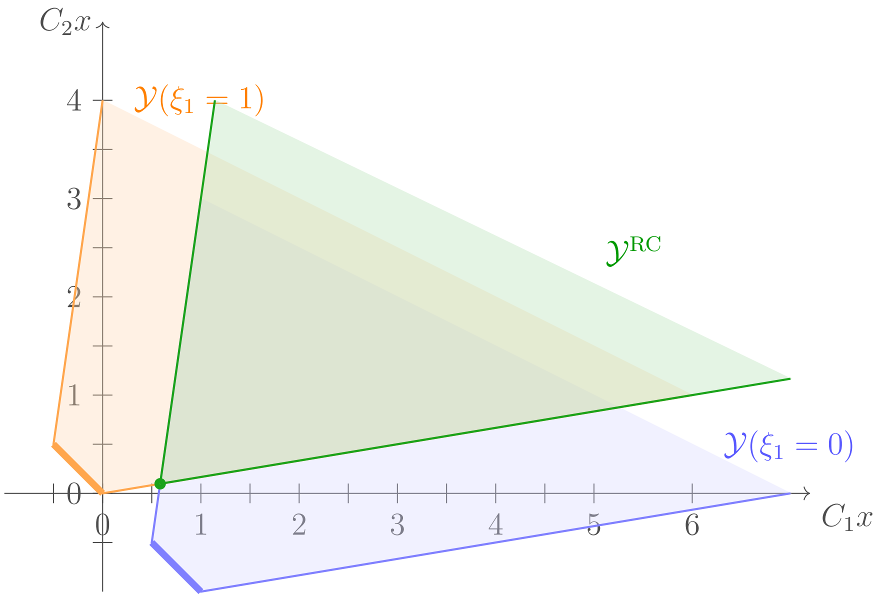


Figure 2.2: The outcome sets for $\xi_1 = 1$ and $\xi_1 = 0$, and for the robust feasible set.

Because $C = I$, we know that $C_1x = x_1$ and $C_2x = x_2$, so $X^{RC} = \mathcal{Y}^{\mathrm{RC}}$ and the robust Pareto set is only a single point, i.e., $\mathcal{Y}_P^{RC} = \{(0.5854, 0.0976)^T\}$.

2.2 Robustness Gap

For the family of uncertain MOLPs of form (2.1), $\inf_{z\in\mathcal{Y}_P^{\mathrm{RC}}} = \min_{z\in\mathcal{Y}_P^{\mathrm{RC}}}$ and $\inf_{y\in\mathcal{Y}_P(\xi)} = \min_{y\in\mathcal{Y}_P(\xi)}$, so we can formulate the robustness gap as

$$\vartheta = \min_{\xi\in\mathcal{U}} \min_{z\in\mathcal{Y}_P^{\mathrm{RC}}} \min_{y\in\mathcal{Y}_P(\xi)} \|z - y\|_p.$$

For this kind of MOLPs, it is also possible to give a condition under which the robustness gap is 0. It turns out, that the robustness gap is zero whenever the Pareto set of the robust counterpart has full dimension.

Proposition 2. Let X^{RC} of (2.1) be given as (2.3). If there exists an $(n-1)$-dimensional face $\mathcal{F}$ of the polyhedral set X^{RC} such that $\mathcal{F} \subseteq X_E^{\mathrm{RC}}$, then

$$\vartheta = 0,$$

and there exists a scenario $\bar{\xi} \in \mathcal{U}$ and an $(n-1)$-dimensional face $\bar{\mathcal{F}}$ of the polyhedral set $X(\bar{\xi})$ such that $\bar{\mathcal{F}} \subseteq X_E(\bar{\xi})$.

Proof. By [3, Cor. 5.7], $\mathcal{F} \subseteq X_E^{\mathrm{RC}}$ if and only if there exist $1 \leq j \leq m$ such that

$$\mathcal{F} = \{x \in \mathbb{R}^n \mid A_j x = \tilde{b}_j,\ A_i x \geq \tilde{b}_i \ \forall 1 \leq i \leq m\},$$

and $\mu \in \mathbb{R}^l_>$ such that

$$A_j = \sum_{i=1}^{l} \mu_i C_i. \tag{2.4}$$

Because $\mathcal{U}$ is compact, it holds that $\inf_{\xi \in \mathcal{U}} \xi = \min_{\xi \in \mathcal{U}} \xi$ and there exists $\bar{\xi} \in \mathcal{U}$ such that $\tilde{b}_j = b_j - \min_{\xi \in \mathcal{U}} A_j \xi = b_j - A_j \bar{\xi}$. For every $x \in \mathcal{F}$ and every $1 \leq i \leq m$ we have $A_i x \geq \tilde{b}_i \geq b_i - A_i \bar{\xi}$ and hence

$$\mathcal{F} \subseteq \bar{\mathcal{F}} := \{x \in \mathbb{R}^n \mid A_j x = \tilde{b}_j,\ A_i x \geq b_i - A_i \bar{\xi} \ \forall 1 \leq i \leq m\} \subseteq X(\bar{\xi}).$$

Applying the reverse direction of [3, Cor. 5.7] to both $\bar{\mathcal{F}}$ and (2.4), we obtain $\bar{\mathcal{F}} \subseteq X_E(\bar{\xi})$ and hence, $\mathcal{F} \subseteq \bar{\mathcal{F}} \subseteq X_E(\bar{\xi})$. Therefore, $\mathcal{F} \subseteq X_E^{\mathrm{RC}} \cap X_E(\bar{\xi})$ and, because $C^{RC}(x) = Cx$ for all $x \in \mathbb{R}^n$ in this chapter, applying (1.10), we obtain $\vartheta = 0$. □

Example 2.2.1. Continuing Example 2.1.1 to find the robustness gap, we want to find the minimal distance between $\mathcal{Y}_P^{RC} = \{(0.5854, 0.0976)^T\}$ and all Pareto points in all scenarios which in our case is the line segment between $(-0.5, 0.5)$ and $(1, -1)$.

In this special case, we can reformulate the problem of finding the gap as

$$\left\{\begin{array}{lrrcl} (P(\xi)) & \vartheta = & \min & \multicolumn{2}{l}{\sqrt{(x_1 - 0.5854)^2 + (x_2 - 0.0976)^2}} \\ & & \text{s. t.} & x_1 + x_2 & = & 0 \\ & & & x_1 & \geqq & -\frac{1}{2} \\ & & & x_1 & \leqq & 1 \end{array}\right\}.$$

This problem can be solved via MATLAB with the function fmincon and leads to a robustness gap of 0.4829 which is illustrated in the following figure.

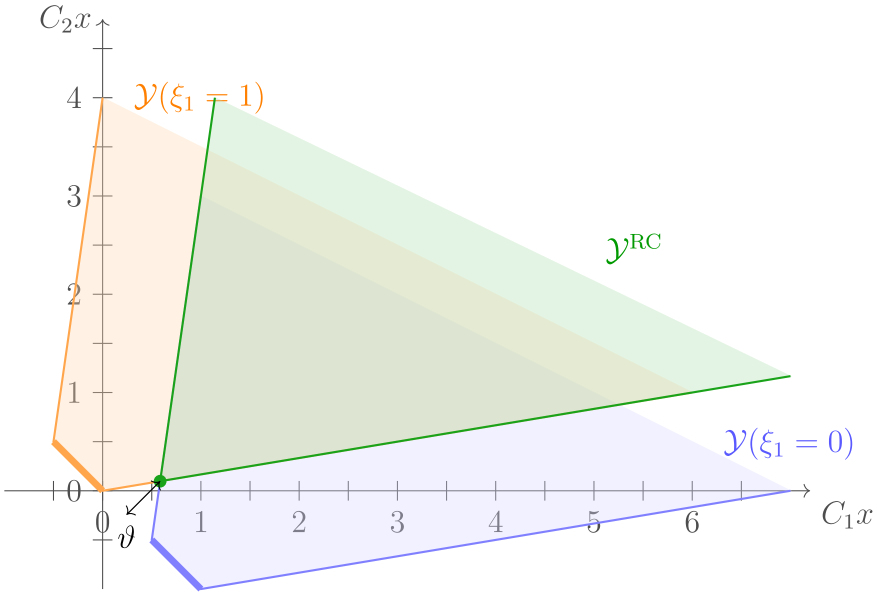


Figure 2.3: Illustration of the robustness gap (which occurs, e.g., together with $\mathcal{Y}(\xi_1 = 0.5)$).

2.3 Bounds

Because computing the gap is very difficult even for the case of MOLPs, we instead compute the upper and lower bounds on the gap.

2.3.1 Lower Bound

To construct a lower bound of the robustness gap for the family of uncertain MOLPs (2.1), we start with the definition for the lower bound of the gap (1.11) and (1.12), use the formulation of the robust counterpart (2.4) and reformulate the problem until it becomes solvable. Recall

$$\Delta^L := \inf_{\xi \in \mathcal{U}} \min_{\substack{\lambda \in \mathbb{R}^p_{\geq}, \\ \|\lambda\|_q = 1}} \big(\min_{z \in \mathcal{Y}^{\mathrm{RC}}} \lambda^T z - \min_{y \in \mathcal{Y}(\xi)} \lambda^T y \big).$$

Based on Chapter 2.1, we rewrite the two minimizations in the parentheses as

$$\min_{z \in \mathcal{Y}^{\mathrm{RC}}} \lambda^T z = \min_{x \in X^{RC}} \lambda^T Cx = \min \left\{ \lambda^T Cx \mid x \in \mathbb{R}^n, Ax \geqq \tilde{b} \right\}$$

and

$$\min_{y \in \mathcal{Y}(\xi)} \lambda^T y = \min_{x \in X(\xi)} \lambda^T Cx = \min \left\{ \lambda^T Cx \mid x \in \mathbb{R}^n, Ax \geqq b - A\xi \right\}.$$

This leads to a a new formulation of Δ^L:

$$\Delta^L = \inf_{\xi \in \mathcal{U}} \min_{\substack{\lambda \in \mathbb{R}^p_{\geq}, \\ \|\lambda\|_q = 1}} \begin{pmatrix} \min\limits_{x} & \lambda^T Cx & & - & \min\limits_{x} & \lambda^T Cx & & \\ \text{s. t.} & Ax & \geqq & \tilde{b} & \text{s. t.} & Ax & \geqq & b - A\xi \end{pmatrix}.$$

Defining $\xi' := b - A\xi$ and using linear programming duality to the right hand side minimization problem, we get

$$\Delta^L = \inf_{\xi \in \mathcal{U}} \min_{\substack{\lambda \in \mathbb{R}^p_{\geqq}, \\ \|\lambda\|_q = 1}} \left(\begin{array}{lllllll} \min\limits_x & \lambda^T Cx & & - & \max\limits_v & v^T \xi' & \\ \text{s. t.} & Ax & \geqq \tilde{b} & & \text{s. t.} & v^T A & = & \lambda^T C \\ & & & & & v & \geqq & 0 \\ & & & & & \xi' & = & b - A\xi \end{array} \right),$$

where v are the dual variables associated with the constraint $Ax \geqq \xi'$. In the parentheses, the minimum depends only on x, while the maximum depends on v, so we can reformulate $\min\limits_x \lambda^T Cx - \max\limits_v v^T \xi'$ into $\min\limits_{x,v} \lambda^T Cx - v^T \xi'$.

Because $\mathcal{U}$ is a compact set, we can replace the infimum by a minimum and get a final problem formulation

$$\Delta^L = \left(\begin{array}{llll} \min\limits_{x,v,\lambda,\xi} & \lambda^T Cx - v^T \xi' & & \\ \text{s. t.} & Ax & \geqq & \tilde{b} \\ & v^T A & = & \lambda^T C \\ & \|\lambda\|_q & = & 1 \\ & \lambda, v & \geqq & 0 \\ & \xi' & = & b - A\xi \\ & \xi & \in & \mathcal{U} \end{array} \right). \tag{2.5}$$

This formulation turns out to become a DC optimization problem if C can, by adding 0-rows, be extended to a quadratic matrix which is positive semidefinite.

Example 2.3.1. Continuing Example 2.2.1 to find the lower bound, we look for a scenario over all possible scenarios for which over all possible nonnegative vectors

$\lambda \in \mathbb{R}^2_{\geq}$ the distance between the Pareto set of this scenario $\mathcal{Y}_P(\xi)$ and the the robust Pareto set $\mathcal{Y}_P^{RC}$ is minimal. In this example, these are scenario ξ such that $\xi_1 = 0$ and vector $\lambda = (1, 0)^T$, which lead to a lower gap of 0.0854.

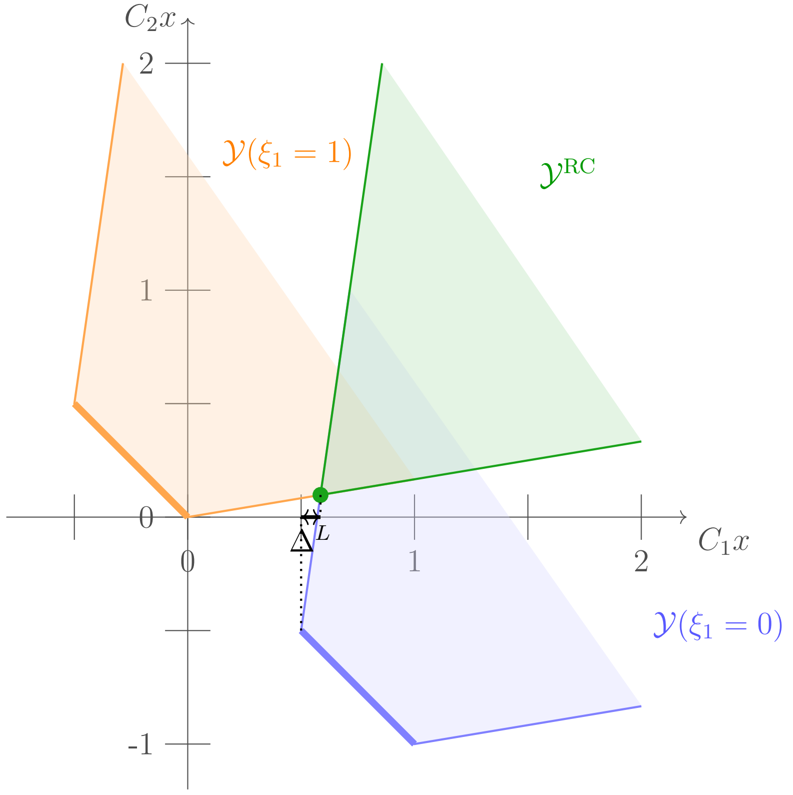


Figure 2.4: Illustration of the lower bound for the robustness gap.

2.3.2 Upper bound

To construct an upper bound on the robustness gap for the family of uncertain MOLPs (2.1), we start with the definition for the upper bound of the gap (1.11) and (1.12), use the formulation of the robust counterpart (2.4) and reformulate the problem until it becomes solvable. Recall

$$\Delta^U := \inf_{\xi\in\mathcal{U}} \max_{\substack{\lambda\in\mathbb{R}^p_{\geqq},\\ \|\lambda\|_q=1}} (\min_{z\in\mathcal{Y}^{\mathrm{RC}}} \lambda^T z - \min_{y\in\mathcal{Y}(\xi)} \lambda^T y).$$

Based on Chapter 2.1, we rewrite the two minimizations in the parentheses as

$$\min_{z\in\mathcal{Y}^{\mathrm{RC}}} \lambda^T z = \min_{x\in X^{RC}} \lambda^T Cx = \min\left\{\lambda^T Cx \mid x\in\mathbb{R}^n, Ax \geqq \tilde{b}\right\},$$

and

$$\min_{y\in\mathcal{Y}(\xi)} \lambda^T y = \min_{x\in X(\xi)} \lambda^T Cx = \min\left\{\lambda^T Cx \mid x\in\mathbb{R}^n, Ax \geqq b - A\xi\right\}.$$

This leads to a new formulation of Δ^U:

$$\Delta^U = \inf_{\xi\in\mathcal{U}} \max_{\substack{\lambda\in\mathbb{R}^p_{\geqq},\\ \|\lambda\|_q=1}} \left(\begin{array}{llllllll} \min\limits_x & \lambda^T Cx & & & - & \min\limits_x & \lambda^T Cx & \\ \text{s. t.} & Ax & \geqq & \tilde{b} & & \text{s. t.} & Ax \quad \geqq & b - A\xi \end{array}\right).$$

Defining $\xi' := b - A\xi$ and using linear programming duality to the left hand side minimization problem, we get

$$\Delta^U = \inf_{\xi\in\mathcal{U}} \max_{\substack{\lambda\in\mathbb{R}^p_{\geqq},\\ \|\lambda\|_q=1}} \left(\begin{array}{lllllllll} \max\limits_u & u^T\tilde{b} & & & - & \min\limits_x & \lambda^T Cx & & \\ \text{s. t.} & u^T A & = & \lambda^T C & & \text{s. t.} & Ax & \geqq & \xi' \\ & u & \geqq & 0 & & & \xi' & = & b - A\xi \end{array}\right),$$

where u are the dual variables associated with the constraint $Ax \geqq \tilde{b}$. In the parentheses, the maximum depends only on u, while the minimum depends only on x, so we can reformulate $\max\limits_u u^T\tilde{b} - \min\limits_x \lambda^T Cx$ into $\max\limits_{x,u} u^T\tilde{b} - \lambda^T Cx$.

Because $\mathcal{U}$ is a compact set, we can replace the infimum by a minimum and get a final problem formulation

$$\Delta^U = \min_{\xi \in \mathcal{U}} \left(\begin{array}{lrcl} \max\limits_{x,u,\lambda} & u^T\tilde{b} - \lambda^T Cx & & \\ \text{s. t.} & Ax & \geqq & \xi' \\ & u^T A & = & \lambda^T C \\ & \|\lambda\|_q & = & 1 \\ & \lambda, u & \geqq & 0 \\ & \xi' & = & b - A\xi \end{array} \right). \tag{2.6}$$

It turns out that even this minimization problem looks similar to the formulation of Δ^L, it can not be easily solved. This is the case because it is a two-stage problem which can not be rewritten in a nicer way.

Example 2.3.2. Continuing Example 2.3.1 to find the upper bound, we look for a scenario where the biggest distance measured as a weighted-sum to our robust Pareto point $\mathcal{Y}_P^{RC}$ is minimal. This is the case for the scenario for which the line segment from $\mathcal{Y}_P^{RC}$ to the midpoint of the Pareto set of this scenario is diagonal to this Pareto set. This is the case for $\xi_1 = 0.5061$ and leads to an upper bound of 0.5915 for either $\lambda = (1, 0)^T$ or $\lambda = (0, 1)^T$.

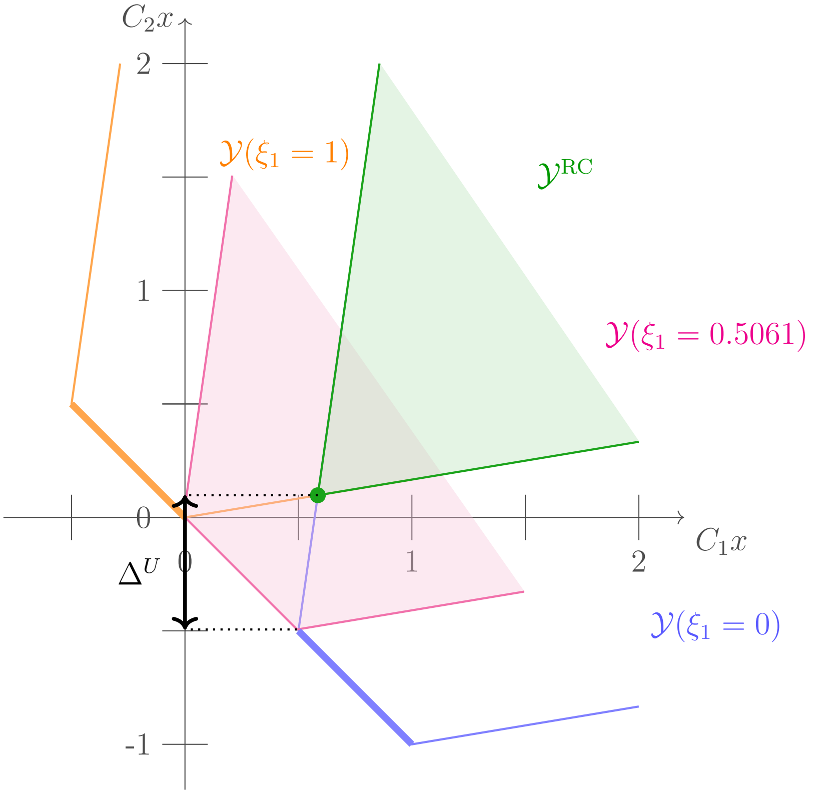


Figure 2.5: Illustration of the upper bound for the robustness gap.

Chapter 3

Parameter Uncertainty

In this chapter, we focus on MOLPs with uncertainty in the data A, b and C. So our uncertainty $\xi = (\eta, C, A, b) \in \mathcal{U}$ can be rewritten as $\xi = (C, A, b) \in \mathcal{U}$, because we do not have decision uncertainty, i.e. $\eta = 0$. This kind of uncertainty occurs when we can not trust the given data because of measurement errors or because the data is influenced by the future, like in Chapter 1.3.

In this thesis, we only consider uncertainty in A and b, because all MOLPs with uncertainty in C can be reformulated to MOLPs with uncertainty only in A and b, see for example [5]. Parameter uncertainty in A and b leads to the following uncertain MOLP:

$$\left\{ \begin{array}{lllll} (\text{MOLP}(\xi)) & \min & Cx & & \\ & \text{s. t.} & Ax & \geqq & b \\ & & x & \in & \mathbb{R}^n \end{array} \right\}_{\xi=(A,b)\in\mathcal{U}} , \tag{3.1}$$

where $\mathcal{U} \subset \mathbb{R}^{m\times n} \times \mathbb{R}^m$.

For this class of MOLPs,

$$X(\xi) = \{x \in \mathbb{R}^n | Ax \geqq b\}.$$

Note also, that MOLPs with decision uncertainty can be reformulated into MOLPs with parameter uncertainty, so Chapter 2 is just a special case of this chapter. As stated in Chapter 1, we assume that $\mathcal{U}$ is compact and convex.

To understand (3.1) better, we analyze the associated robust counterpart and present formulations of the optimization problems whose optimal solutions determine the upper and lower bounds on the robustness gap.

3.1 Robust Counterpart

To find the robust counterpart to the family of MOLPs of form (3.1), we identify the robust feasible set defined as the set of decision vectors which are feasible for all possible uncertainty vectors in $\mathcal{U}$, namely

$$X^{\mathrm{RC}} := \{x \in \mathbb{R}^n \mid Ax \geqq b, \ \forall \xi = (A, b) \in \mathcal{U}\}.$$

Because $\mathcal{U}$ is a compact and convex set, there exists a finite representation of the constraints, which we denote as $(\tilde{A}, \tilde{b}) \in \mathbb{R}^{r \times n} \times \mathbb{R}^r$ where $(\tilde{A}_i, \tilde{b}_i) = \tilde{\xi}_i$ is one component of a scenario in $\mathcal{U}$. This leads to a new formula for X^{RC}:

$$X^{RC} = \left\{x \in \mathbb{R}^n \mid \tilde{A}x \geqq \tilde{b}\right\} \tag{3.2}$$

and we can compute the robust counterpart of the parameter uncertainty problem after finding $(\tilde{A}, \tilde{b})$. The difficulty to find $(\tilde{A}, \tilde{b})$ depends on the structure of $\mathcal{U}$.

Unfortunately, it is not as easy as in Chapter 2 to compute $(\tilde{A}, \tilde{b})$, because $\tilde{A}$ and $\tilde{b}$ have to be computed simultaneous and depend on each other. With this finite formulation (3.2), we can formulate the robust counterpart as a deterministic MOLP with finite constraints:

$$\left(\begin{array}{lllll} \left(RC^{MOLP_{Parameter}}\right) & \min & Cx & & \\ & \text{s. t.} & \tilde{A}x & \geqq & \tilde{b} \\ & & x & \in & \mathbb{R}^n \end{array}\right).$$

3.2 Robustness Gap and Bounds

For the family of uncertain MOLPs of form (2.1), $\inf_{z \in \mathcal{Y}_P^{\mathrm{RC}}} = \min_{z \in \mathcal{Y}_P^{\mathrm{RC}}}$ and $\inf_{y \in \mathcal{Y}_P(\xi)} = \min_{y \in \mathcal{Y}_P(\xi)}$, so we can formulate the robustness gap as

$$\vartheta = \min_{\xi \in \mathcal{U}} \min_{z \in \mathcal{Y}_P^{\mathrm{RC}}} \min_{y \in \mathcal{Y}_P(\xi)} \|z - y\|_p.$$

3.2.1 Bounds

Because computing the gap is very difficult even for the case of MOLPs, we instead compute upper and lower bounds on the gap.

3.2.1.1 Lower Bound

To construct a lower bound of the robustness gap for MOLPs of the form (3.1), we start with the definition for the lower bound of the gap (1.11) and (1.12), use the formulation of the robust counterpart (3.2) and reformulate the system until it becomes solvable. Recall

$$\Delta^L := \inf_{\xi\in\mathcal{U}} \min_{\substack{\lambda\in\mathbb{R}^p_{\geq},\\ \|\lambda\|_q=1}} \big(\min_{z\in\mathcal{Y}^{\mathrm{RC}}} \lambda^T z - \min_{y\in\mathcal{Y}(\xi)} \lambda^T y).$$

Based on Chapter 3.1, we rewrite the two minimizations in the parentheses as

$$\min_{z\in\mathcal{Y}^{\mathrm{RC}}} \lambda^T z = \min_{x\in X^{RC}} \lambda^T Cx = \min\left\{\lambda^T Cx \mid x\in\mathbb{R}^n, \tilde{A}x \geqq \tilde{b}\right\}$$

and

$$\min_{y\in\mathcal{Y}(\xi)} \lambda^T y = \min_{x\in X(\xi)} \lambda^T Cx = \min\left\{\lambda^T Cx \mid x\in\mathbb{R}^n, Ax \geqq b\right\}.$$

This leads to a a new formulation of Δ^L:

$$\Delta^L = \inf_{\xi=(A,b)\in\mathcal{U}} \min_{\substack{\lambda\in\mathbb{R}^p_{\geq},\\ \|\lambda\|_q=1}} \left(\begin{array}{llllllll} \min\limits_{x} & \lambda^T Cx & & & - & \min\limits_{x} & \lambda^T Cx & \\ \text{s. t.} & \tilde{A}x & \geqq & \tilde{b} & & \text{s. t.} & Ax \geqq & b \end{array} \right).$$

Applying linear programming duality to the right hand side minimization problem, we get

$$\Delta^L = \inf_{\xi=(A,b)\in\mathcal{U}} \min_{\substack{\lambda\in\mathbb{R}^p_{\geq},\\ \|\lambda\|_q=1}} \left(\begin{array}{lllllllll} \min\limits_{x} & \lambda^T Cx & & & - & \max\limits_{v} & v^T\tilde{b} & & \\ \text{s. t.} & \tilde{A}x & \geqq & \tilde{b} & & \text{s. t.} & v^T\tilde{A} & = & \lambda^T C \\ & & & & & & v & \geqq & 0 \end{array} \right),$$

where v are the dual variables associated with the constraint $Ax \geqq b$. In the parentheses, the minimum depends only on x, while the maximum depends on v, so we can reformulate our $\min\limits_{x} \lambda^T Cx - \max\limits_{v} v^T\tilde{b}$ into $\min\limits_{x,v} \lambda^T Cx - v^T\tilde{b}$.

Because $\mathcal{U}$ is a compact set, we can replace the infimum by a minimum and get a final problem formulation

$$\Delta^L = \left(\begin{array}{lrcl} \min\limits_{x,v,\lambda,\xi=(A,b)} & \lambda^T Cx - v^T \tilde{b} & & \\ \text{s. t.} & \tilde{A}x & \geqq & \tilde{b} \\ & v^T \tilde{A} & = & \lambda^T C \\ & \|\lambda\|_q & = & 1 \\ & \lambda, v & \geqq & 0 \\ & \xi = (A,b) & \in & \mathcal{U} \end{array} \right). \tag{3.3}$$

This formulation turns out to become a DC optimization problem if C can, by adding 0-rows, be extended to a quadratic matrix which is positive semidefinite.

3.2.1.2 Upper Bound

To construct an upper bound on the robustness gap for the family of uncertain MOLPs (3.1), we start with the definition for the upper bound of the gap (1.11) and (1.12), use the formulation of the robust counterpart (3.2) and reformulate the problem until it becomes solvable. Recall

$$\Delta^U := \inf_{\xi \in \mathcal{U}} \max_{\substack{\lambda \in \mathbb{R}^p_{\geq}, \\ \|\lambda\|_q = 1}} (\min_{z \in \mathcal{Y}^{\text{RC}}} \lambda^T z - \min_{y \in \mathcal{Y}(\xi)} \lambda^T y).$$

Based on Chapter 3.1, we rewrite the two minimizations in parentheses as

$$\min_{z \in \mathcal{Y}^{\text{RC}}} \lambda^T z = \min_{x \in X^{RC}} \lambda^T Cx = \min \left\{ \lambda^T Cx \mid x \in \mathbb{R}^n, \tilde{A}x \geqq \tilde{b} \right\}$$

and

$$\min_{y \in \mathcal{Y}(\xi)} \lambda^T y = \min_{x \in X(\xi)} \lambda^T Cx = \min \left\{ \lambda^T Cx \mid x \in \mathbb{R}^n, Ax \geqq b \right\}.$$

This leads to a a new formulation of Δ^U:

$$\Delta^U = \inf_{\xi=(A,b)\in\mathcal{U}} \max_{\substack{\lambda \in \mathbb{R}^p_{\geqq}, \\ \|\lambda\|_q = 1}} \begin{pmatrix} \min\limits_x & \lambda^T Cx & & - & \min\limits_x & \lambda^T Cx & & \\ \text{s. t.} & \tilde{A}x & \geqq & \tilde{b} & \text{s. t.} & Ax & \geqq & b \end{pmatrix}.$$

Applying linear programming duality to the left hand side minimization problem, we get

$$\Delta^U = \inf_{\xi=(A,b)\in\mathcal{U}} \max_{\substack{\lambda \in \mathbb{R}^p_{\geqq}, \\ \|\lambda\|_q = 1}} \begin{pmatrix} \max\limits_u & u^T\tilde{b} & & & - & \min\limits_x & \lambda^T Cx & & \\ \text{s. t.} & u^T\tilde{A} & = & \lambda^T C & & \text{s. t.} & \tilde{A}x & \geqq & \tilde{b} \\ & u & \geqq & 0 & & & & & \end{pmatrix},$$

where u are the dual variables associated with the constraint $\tilde{A}x \geqq \tilde{b}$. In parentheses, the maximum depends only on u, while the minimum depends only on x, so we can reformulate our $\max\limits_u u^T\tilde{b} - \min\limits_x \lambda^T Cx$ into $\max\limits_{x,u} u^T\tilde{b} - \lambda^T Cx$.

Because $\mathcal{U}$ is a compact set, we can replace the infimum by a minimum and get a final problem formulation

$$\Delta^U = \min_{\xi=(A,b)\in\mathcal{U}} \left(\begin{array}{lrcl} \max\limits_{x,u,\lambda} & u^T\tilde{b} - \lambda^T Cx & & \\ \text{s. t.} & \tilde{A}x & \geqq & \tilde{b} \\ & u^T\tilde{A} & = & \lambda^T C \\ & \|\lambda\|_q & = & 1 \\ & \lambda, u & \geqq & 0 \end{array} \right). \tag{3.4}$$

Similar to Chapter 2, this problem cannot be solved as easily as the lower bound Δ^L.

Chapter 4

Numerical Experiments

In this section we examine three different classes of examples with different uncertainty sets $\mathcal{U}$ and analyze numerical issues that accompany the computations.

All examples use the same C, A and b, namely

$$C = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad A = \begin{pmatrix} -\frac{1}{6} & 1 \\ 7 & -1 \\ 1 & 1 \end{pmatrix}, \quad b = \begin{pmatrix} -\frac{1}{6} \\ 3 \\ 1 \end{pmatrix}.$$

In order to compute the upper and lower bounds for the robustness gap, it is required that optimization problems (2.4) and (2.5) be solved. Problem (2.4) for the lower bound Δ^L can be solved with the following methods:

- **one-stage problem** in the form of (2.4).
- **two-stage problem**

  Discretize $\mathcal{U}$ and minimize with respect to x, v and λ for a fixed $\xi \in \mathcal{U}$. Choose the smallest value $\Delta^L(\xi)$ over all $\xi \in \mathcal{U}_{discrete}$.

- **three-stage problem**

  Discretize $\mathcal{U}$ and $\Lambda = \{\lambda \in \mathbb{R}^n_{\geq} |\ ||\lambda||_q = 1\}$ and minimize with respect to x and v for fixed $\xi \in \mathcal{U}$ and $\lambda \in \Lambda$. Choose the smallest value $\Delta^L(\xi, \lambda)$ over all $\xi \in \mathcal{U}_{discrete}$ and $\lambda \in \Lambda_{discrete}$.

In the examples we solve, $\mathcal{Y}_P^{RC}$ is a singleton we can calculate if the gap is unequal to zero. Based on Theorem 1, the point $\bar{x}$ for the optimal solution of the weighted-sum scalarization is in the Pareto set, so we can reformulate (2.4) in this special case as

$$\Delta^L = \left(\begin{array}{lrcl} \min\limits_{v,\lambda,\xi} & \multicolumn{3}{l}{\lambda^T C\bar{z} - v^T\xi'} \\ \text{s. t.} & v^T A & = & \lambda^T C \\ & \|\lambda\|_q & = & 1 \\ & \lambda, v & \geqq & 0 \\ & \xi' & = & b - A\xi \\ & \xi & \in & \mathcal{U} \end{array} \right), \tag{4.1}$$

where $\bar{z}$ is the Pareto point of the robust counterpart. In effect, we get three more methods for computing the lower bound. They work similarly to the three methods above, except that they use the new formulation (4.1) of Δ^L instead of (2.4):

- **one-stage problem with a fixed Pareto point** $\bar{z}$

- **two-stage problem with a fixed Pareto point** $\bar{z}$

- **three-stage problem with a fixed Pareto point** $\bar{z}$

Wile the lower bound is a minimization problem, the upper bound (2.5) is a min-max problem depending on different variables. This can not be solved easily as

a one-stage problem. So instead of solving it directly, we have to dicretize $\mathcal{U}$ and perform different iterations of the inner part of the problem for fixed $\xi \in \mathcal{U}_{discrete}$ to get a bound. Because we are looking for the minimum of all the solutions of the inner part, this is also giving us an upper bound for the robustness gap which is greater or equal to the real Δ^U. The quality of this bound can depend on the discretization of $\mathcal{U}$. Based on the special form of the union of $X_E(\xi)$ as a Minkowski difference and the chosen $C = I$, we only have to discretize the extreme values of $\mathcal{U}$.
In a similar way as for the lower bound, we get four possible methods with discretizing $\mathcal{U}$ and possibly Λ with the difference of taking the maximum with respect to all $\lambda \in \Lambda$. The methods are:

- **two-stage problem**
- **three-stage problem**
- **two-stage problem with a fixed Pareto point** $\bar{z}$
- **three-stage problem with a fixed Pareto point** $\bar{z}$

A big concern about solving multi-stage problems is that the computation takes a lot of time. Depending on the number of discretization points, the time can increase significantly.

A pseudo-code for the two-stage problem to compute the lower bound is given in Algorithm 1. The algorithm for the upper bound is similar, except we use Δ^U instead of Δ^L and change the 6th row of the pseudo-code into

$$\max_{x,u,\lambda} u^T\tilde{b} - \lambda^T Cx, \text{ s.t. } \xi' = b - A\xi;\ Ax \geqq \xi';\ u^T A = \lambda^T C;\ ||\lambda||_* = 1;\ \lambda, u \geqq 0.$$

In Algorithm 2, a pseudo-code for the three-stage problem to compute the upper

Algorithm 1 computing Δ^L

```
procedure TWO-STAGE $\Delta^L$(matrices $C, A, b$ and $\mathcal{U}_{discrete}$)
    set $\Delta^L = \infty$
    for $i = 1, \ldots m$ do
        compute $\tilde{b}_i = \max_\xi b_i - A_i\xi$, s.t. $\xi \in \mathcal{U}$
    for $\forall \xi \in \mathcal{U}_{discrete}$ do
        compute $\Delta^L(\xi) = \min_{x,v,\lambda} \lambda^T Cx - v^T\xi',$ s.t. $\xi' = b - A\xi;\ Ax \geqq \tilde{b};\ v^T A = \lambda^T C;\ ||\lambda||_* = 1;\ \lambda, v \geqq 0$
        if $\Delta^L(\xi) < \Delta^L$ then
            $\Delta^L = \Delta^L(\xi)$
            save computed $\lambda$ as $\bar{\lambda}$
            save used $\xi$ as $\bar{\xi}$
    return $\Delta^L, \bar{\lambda}, \bar{\xi}$
```

bound is presented.

It turns out that for the class of problems we consider in this chapter, Algorithm 1 correctly solves the two-stage problem for the lower bound without stopping at local minima, while for the upper bound we receive the correct results when we either use the three-stage problem or the two-stage problem with well fitting starting points. Because the three-stage problem can be solved without having further knowledge about good starting points, we decide to choose this method. In the subsequent sections we present the numerical experiments we conducted. All algorithms are implemented in MATLAB using the fmincon function and part of this code is given in the Appendix.

Algorithm 2 computing Δ^U

```
procedure THREE-STAGE $\Delta^U$(matrices $C, A, b, \mathcal{U}_{discrete}$ and $\Lambda_{discrete}$)
    set $\Delta^U = \infty$
    for $i = 1, \dots m$ do
        compute $\tilde{b}_i = \max_\xi b_i - A_i\xi$, s.t. $\xi \in \mathcal{U}$
    for $\forall \xi \in \mathcal{U}_{discrete}$ do
        set $\Delta^U(\xi)$=0
        for $\forall \lambda \in \Lambda_{discrete}$ do
            compute $\Delta^U(\xi, \lambda) = \max_{x,u} u^T\tilde{b} - \lambda^T Cx$, s.t. $\xi' = b - A\xi$; $Ax \geqq \xi'$; $u^T A = \lambda^T C$; $||\lambda||_* = 1$; $\lambda, u \geqq 0$
            if $\Delta^U(\xi, \lambda) \geq \Delta^U(\xi)$ then
                $\Delta^U(\xi) = \Delta^U(\xi, \lambda)$
                save used $\lambda$
        if $\Delta^U(\xi) < \Delta^U$ then
            $\Delta^U = \Delta^U(\xi)$
            save used $\lambda$ as $\bar{\lambda}$
            save used $\xi$ as $\bar{\xi}$
    return $\Delta^U, \bar{\lambda}, \bar{\xi}$
```

4.1 Sensitivity of the Gap with respect to Translation

Let the uncertainty set be defined as $\mathcal{U} := \{\xi \in \mathbb{R}^2 \mid \xi_1 + \xi_2 = c,\ |\xi_1 - \xi_2| \leq 1\}$ where $-50 \leq c \leq 50$, which makes $\mathcal{U}$ be a line segment of the same length for every c, where c models the translation of the segments. From this definition we obtain $\frac{c-1}{2} \leq \xi_1 \leq \frac{c+1}{2}$. For $c = 1$, this is equivalent to the example in Chapter 2. Figure 4.1 depicts the outcome sets $Y(\xi)$ for the two extreme values of ξ_1 that result from the definition of $\mathcal{U}$. The outcome sets $Y(\xi)$ for all other values of ξ_1 are located in $\mathbb{R}^2$ in such a way that the union of their Pareto sets is contained in the line segment from the point $(-\frac{c}{2}, -\frac{c}{2} + 1)$ to the point $(-\frac{c}{2} + 1.5, -\frac{c}{2} - 0.5)$. The robustness gap ϑ does not change as c changes and it can be calculated as the minimal distance between

$\mathcal{Y}_P^{\mathrm{RC}}$ being a singleton and this line segment. We obtain

$$\vartheta = 0.4829.$$

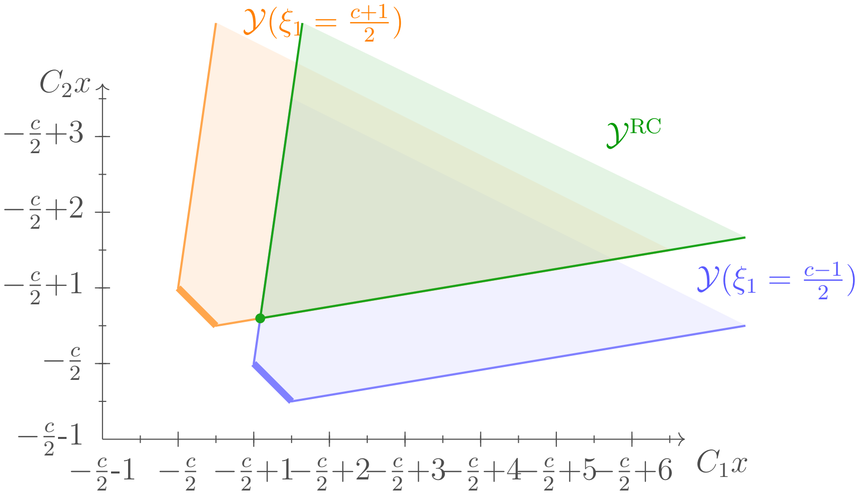


Figure 4.1: The outcome sets $Y(\xi)$ for $\xi_1 = \frac{c-1}{2}$ and $\xi_1 = \frac{c+1}{2}$, and the robust outcome set Y^{RC} for a fixed c.

Below we compare the outcomes of the different methods for computing the bounds as described before.

Computing the lower bound via the one-stage problem and the upper bound via the two-stage problem leads to the bounds in Figure 4.2. We see that the computed lower bound is not correct for negative c-values, because we can evaluate a smaller bound by choosing scenario ξ where $\xi_1 = \frac{c-1}{2}$ with $\lambda = (1,0)^T$. This can happen because the used MATLAB-function can only find local minima of the problem (2.4) and therefore compute the lower bound with $\xi_1 = \frac{c+1}{2}$ and $\lambda = (0,1)$. Also, some of the values for the upper bound are wrong, because they are below the values of the gap. This also happens because fmincon stops at local maxima of (2.5) instead of finding a global maximum.

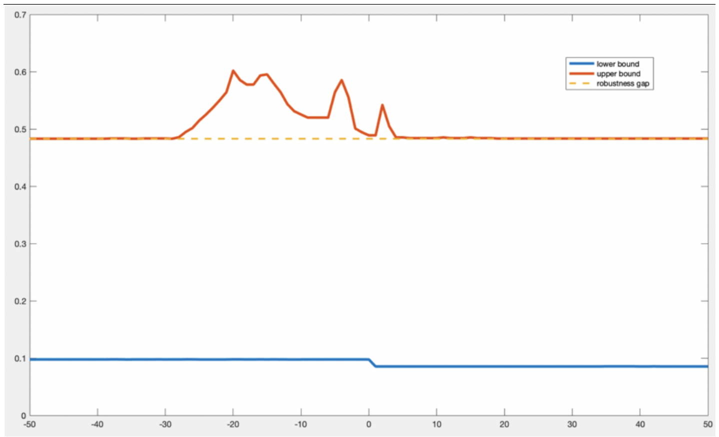


Figure 4.2: Robustness gap and incorrect upper and lower bounds for different c-values.

To fix these problems, we recompute the bound and use the two-stage problem via Algorithm 1 for the lower bound and the three-stage problem vial Algorithm 2 for the upper bound. These computations are reflected in Figure 4.3. This time the lower and upper bound are correct (except for some small computational errors based on the discretization). However, the running time of the used algorithms now is more than 40 times the running time of the algorithms used before. This means, for example, that the computations now take more than 3 hours while they took less than 5 minutes before.

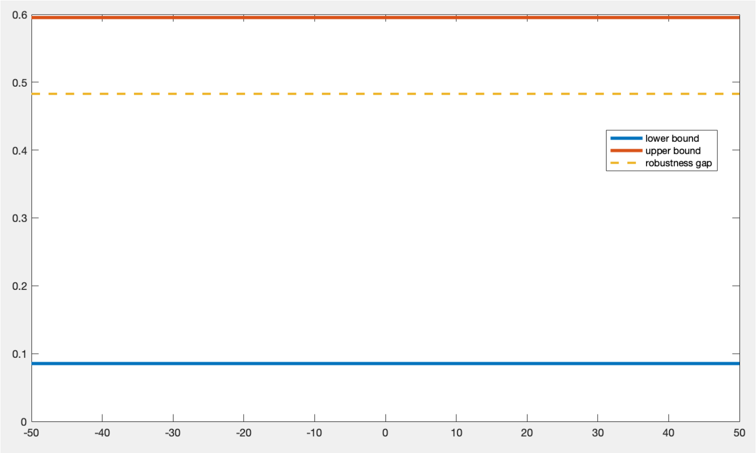


Figure 4.3: Robustness gap and upper and lower bounds for different c-values.

Figure 4.3 depicts the optimal objective values, Δ^L and Δ^U, of problems (2.4) and (2.5), respectively, obtained for $c = -50, -49, \ldots, 49, 50$. For each c we have

$$\Delta^L = 0.0854, \quad \Delta^U = 0.591.$$

The optimal solution $\bar{\xi} = (\frac{c-1}{2}, \frac{c+1}{2})^T$ and $\bar{\lambda} = (1, 0)^T$ yields Δ^L, while Δ^U comes from the optimal solution $\bar{\xi} = (\frac{c-0.0122}{2}, \frac{c+0.0122}{2})^T$ and either $\bar{\lambda} = (1, 0)^T$ or $\bar{\lambda} = (0, 1)^T$. These results confirm that the gap and bounds stay constant under the translation.

This easy example shows, that computing the bound for large problems with many variables in a short time may not be an easy task. To ensure that the computed bounds are correct, it is better to use small discretization steps and allow a long computing time, or use special algorithms for DC-optimization.

4.2 Sensitivity of the Gap with respect to Expansion

Let $\mathcal{U} := \{\xi \in \mathbb{R}^2 \mid \xi_1 + \xi_2 = 1,\ |\xi_1 - \xi_2| \leq c\}$ where $1 \leq c \leq 10$. In this case, $\mathcal{U}$ is a collection of line segments of different length located on the same line in $\mathbb{R}^2$, while c controls the length of the line segments or their expansion. From this definition we obtain $\frac{-c+1}{2} \leq \xi_1 \leq \frac{c+1}{2}$.

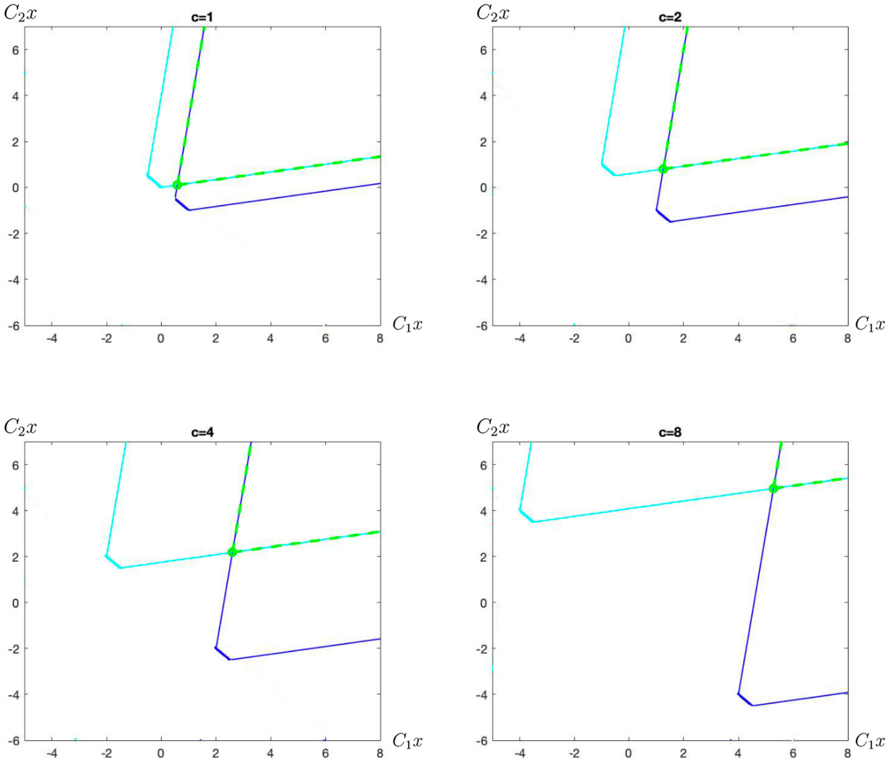


Figure 4.4: The outcome sets $Y(\xi)$ for $\xi_1 = \frac{-c+1}{2}$ and $\xi_1 = \frac{c+1}{2}$, and the robust outcome set Y^{RC} (dotted lines) for different c-values.

Figure 4.4 depicts the outcome sets $Y(\xi)$ for the two extreme values of ξ_1 that result from the definition of $\mathcal{U}$ and for four different values of c. For each c, the outcome sets $Y(\xi)$ for all other values of ξ_1 are located in $\mathbb{R}^2$ in a similar fashion as the outcome sets in Section 4.1.

The robustness gap ϑ can be calculated as the minimal distance between $\mathcal{Y}_P^{\mathrm{RC}}$ being a singleton and the line segment from the point $(\frac{-c+1}{2} - 0.5, \frac{c+1}{2} - 0.5)$ to the point $(\frac{c+1}{2}, \frac{-c+1}{2} - 1)$. As c increases, the gap increases linearly,

$$\vartheta = 0.9658c - 0.4829,$$

which is depicted with the dotted line in Figure 4.5. Problems (2.4) and (2.5) are solved for $c = 1, 1.5, \ldots, 9.5, 10$ and produce the bounds

$$\Delta^L = 0.1707c - 0.0854, \quad \Delta^U = \sqrt{\vartheta^2 + \frac{1}{16}},$$

which are also depicted in Figure 4.5. The optimal solution $\bar{\xi} = (\frac{c+1}{2}, \frac{-c+1}{2})^T$ and $\bar{\lambda} = (1,0)^T$ yields Δ^L and the optimal solution $\bar{\xi} = (0.5061, 0.4939)^T$ and either $\bar{\lambda} = (1,0)^T$ or $\bar{\lambda} = (0,1)^T$ yields Δ^U.

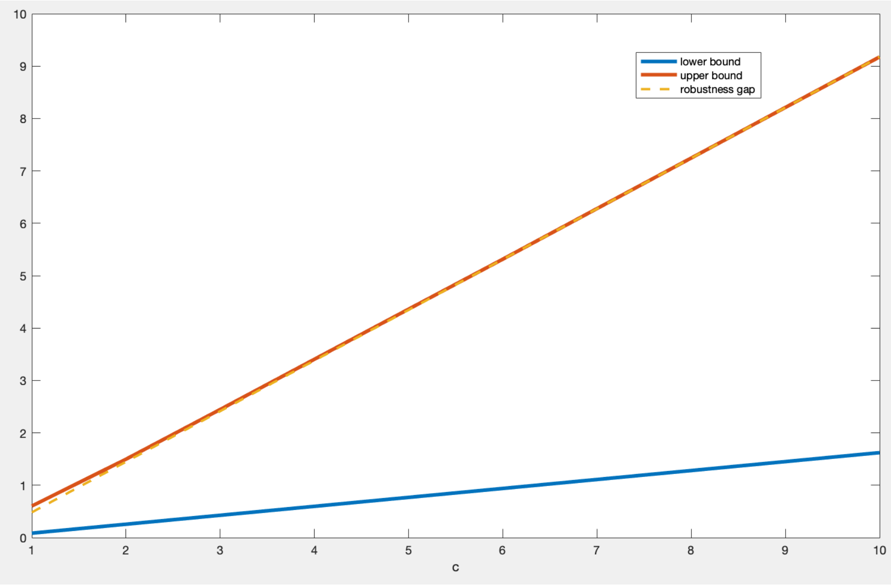


Figure 4.5: The robustness gap and upper and lower bounds for different c-values.

Based on these results we conclude that the robustness gap ϑ and the lower bound Δ^L increase linearly, but the upper bound does not. As c increases, Δ^U approaches the robustness gap ϑ, while the distance between Δ^L and ϑ becomes bigger. This shows that the ratio of each bound and the gap is not constant under the expansion.

4.3 Sensitivity of the Gap with respect to the Norm

In this example, $\mathcal{U} := \{\xi \in \mathbb{R}^2 \mid \|\xi_1 + \xi_2\|_p \leq 1\}$ where $1 \leq p \leq 6$. Figure 4.6 depicts the unit balls of some norms, the outcome sets $Y(\xi)$ for selected extreme vectors $\xi \in \mathcal{U}$, and the robust outcome sets Y^{RC}. The boundaries of the unit balls and the Pareto sets intersect at the midpoints of the Pareto sets.

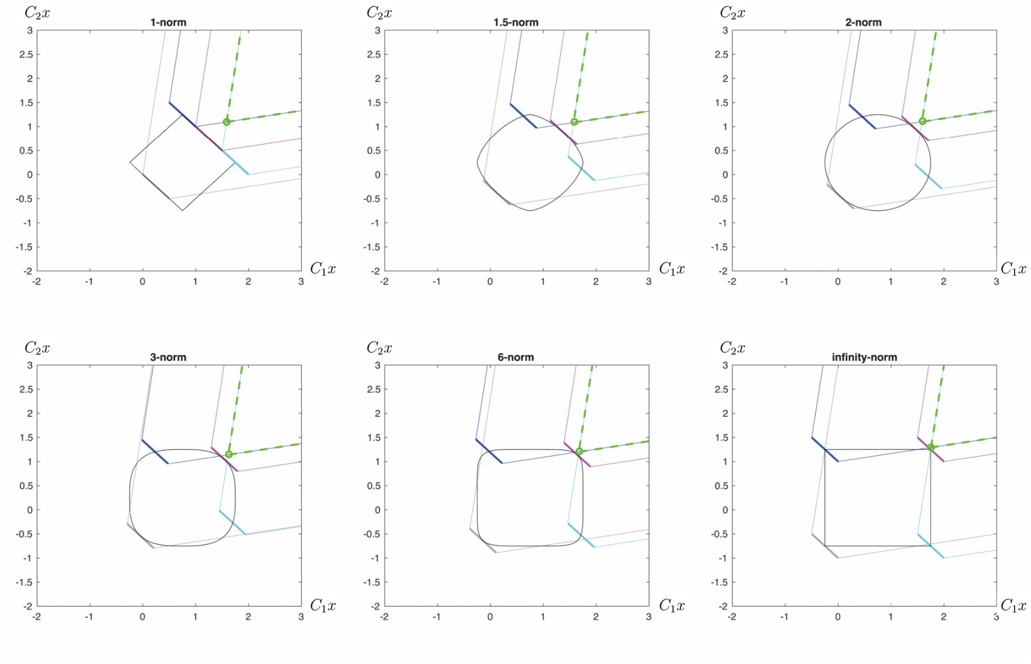


Figure 4.6: The outcome sets $Y(\xi)$ for selected extreme $\xi \in \mathcal{U}$ and the robust outcome set Y^{RC} (dotted lines) for different p-norms.

For every p-norm, the robustness gap ϑ can be calculated as the minimal distance between $\mathcal{Y}_P^{\text{RC}}$ being a singleton and the Pareto set $Y_P(\bar{\xi})$ (plotted in purple), where $\bar{\xi} = \sqrt[p]{\frac{1}{2}}(-1,-1)^T$, and these values are plotted in Figure 4.7 (dotted curve). Problems (2.4) and (2.5) are solved for $p = 1, 1.25, \dots, 5.75, 6$ and the resulting bounds, Δ^L and Δ^U, are plotted in the same figure.

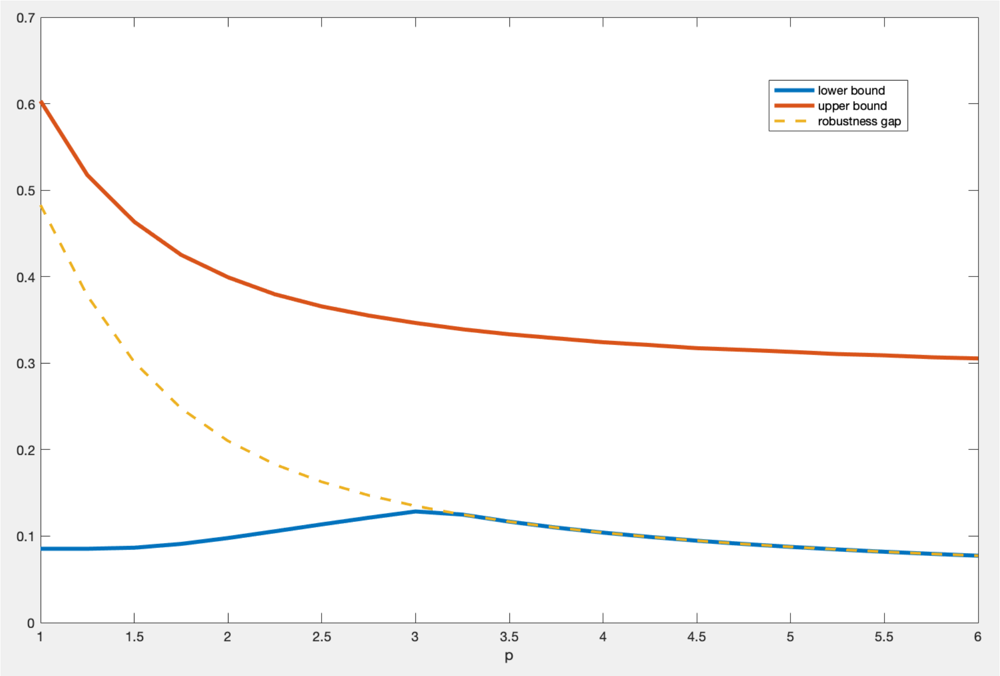


Figure 4.7: The robustness gap and upper and lower bounds for different p-norms.

It is interesting to see a bump in the lower bound for $p \in (2.5; 3.5)$, which can also be examined in Figure 4.8 reporting more numerical results. In this figure, the values of both bounds for different p-norms are accompanied by the associated optimal solutions $(\bar{\xi}_1, \bar{\xi}_2)$ and optimal $\bar{\lambda}_1$. We observe that for some $p^* \in (3, 3.25)$, the optimal solution $\bar{\lambda} = (1,0)^T$ changes to $\bar{\lambda} = \frac{1}{\sqrt{2}}(1,1)^T$ and the optimal solution $(\bar{\xi}_1, \bar{\xi}_2)$ also significantly changes, which is likely to cause the jump. At the same time, the lower bound becomes and remains tight. This is due to the fact, that while $\min_{\substack{\lambda \in \mathbb{R}^2_{\geq}, \\ \|\lambda\|_2 = 1}} \Delta\left(\xi = (-1,0)^T, \lambda\right)$ is increasing when p is increasing, $\min_{\substack{\lambda \in \mathbb{R}^2_{\geq}, \\ \|\lambda\|_2 = 1}} \Delta\left(\xi = \sqrt[p]{\frac{1}{2}}(-1,-1)^T, \lambda\right) = \vartheta$ is decreasing if p is increasing. So the lower bound is computed as $\min_{\substack{\lambda \in \mathbb{R}^2_{\geq}, \\ \|\lambda\|_2 = 1}} \Delta\left(\xi = (-1,0)^T, \lambda\right)$ for all $p < p^*$ and as $\min_{\substack{\lambda \in \mathbb{R}^2_{\geq}, \\ \|\lambda\|_2 = 1}} \Delta\left(\xi = \sqrt[p]{\frac{1}{2}}(-1,-1)^T, \lambda\right) = \vartheta$ for all $p > p^*$.

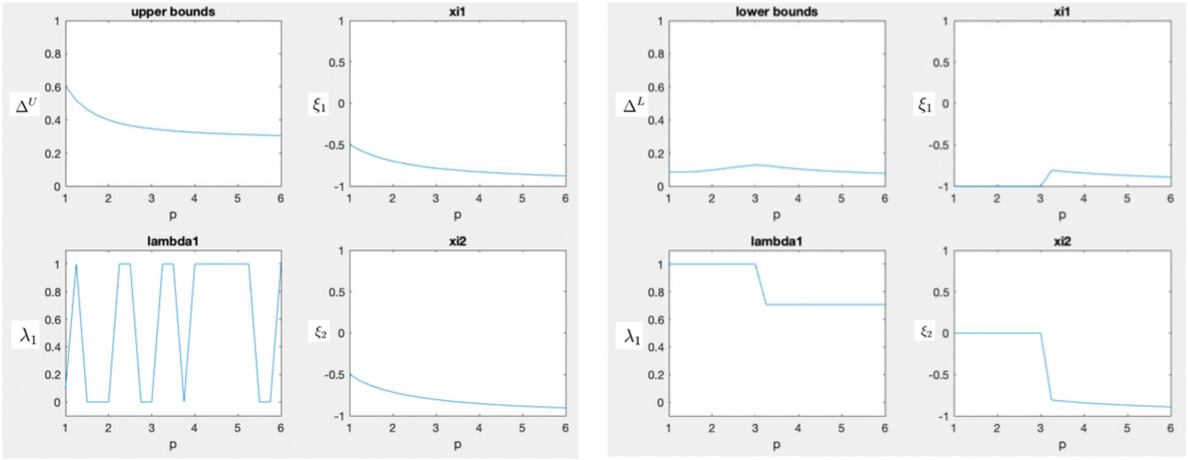


Figure 4.8: Upper and lower bounds, optimal $\bar{\lambda}_1$, and optimal $(\bar{\xi_1}, \bar{\xi_2})$ for different p-norms.

Chapter 5

Conclusion

5.1 Summary

The robustness gap for uncertain multiobjective linear optimization and the upper and lower bounds for this gap are investigated in detail. A real-life uncertain biobjective LP is developed to illustrate the application of the gap in decision making.

Biobjective examples for different sets of decision uncertainty give insight into how the gap and the bounds react to different mathematical transformations.

Even for this biobjective linear case, where the optimization problems for computing the upper and lower bound are formulated, it is difficult to compute these bounds in practice, because these optimization problems have non-convex objective functions assuming local minima or maxima. This kind of problems is not solvable with commonly used minimization functions like fmincon in MATLAB, because those functions often find local extrema rather than global. Therefore, the optimization problems have to be reformulated into multi-stage problems using discretization.

The computed results also show a big range of the ratio between the bounds and the actual robustness gap. There are cases where these bounds are tight, but also cases where the ratio between gap and a bound goes to infinity.

In the real-life application, the bounds on the robustness gap are interesting not only because they give an interval in which the gap is contained, but also because they give more insight into the uncertain multiobjective linear program. The bounds are computed using weighted-sum scalarization problems where the used λ-vectors can serve as a reference for a decision.

5.2 Further work

In the near future, it could be interesting to compute the bounds for MOLPs with decision uncertainty in higher dimensions. It might be possible to fix the issue of the local extrema or shorten the long computational times by using a special algorithm for DC optimization.

The next step task could be to construct examples for MOLPs with Parameter Uncertainty which do not have trivial gaps. There could also exist a proposition giving conditions under which the gap is zero.

It would also be helpful to develop conditions under which the bounds are tight or to find out how far each bound is away from the gap based on the optimal solutions $\{\bar{\xi}, \bar{\lambda}, \bar{x}, (\bar{v} \text{ or } \bar{u})\}$ of the optimization problems yielding these bounds.

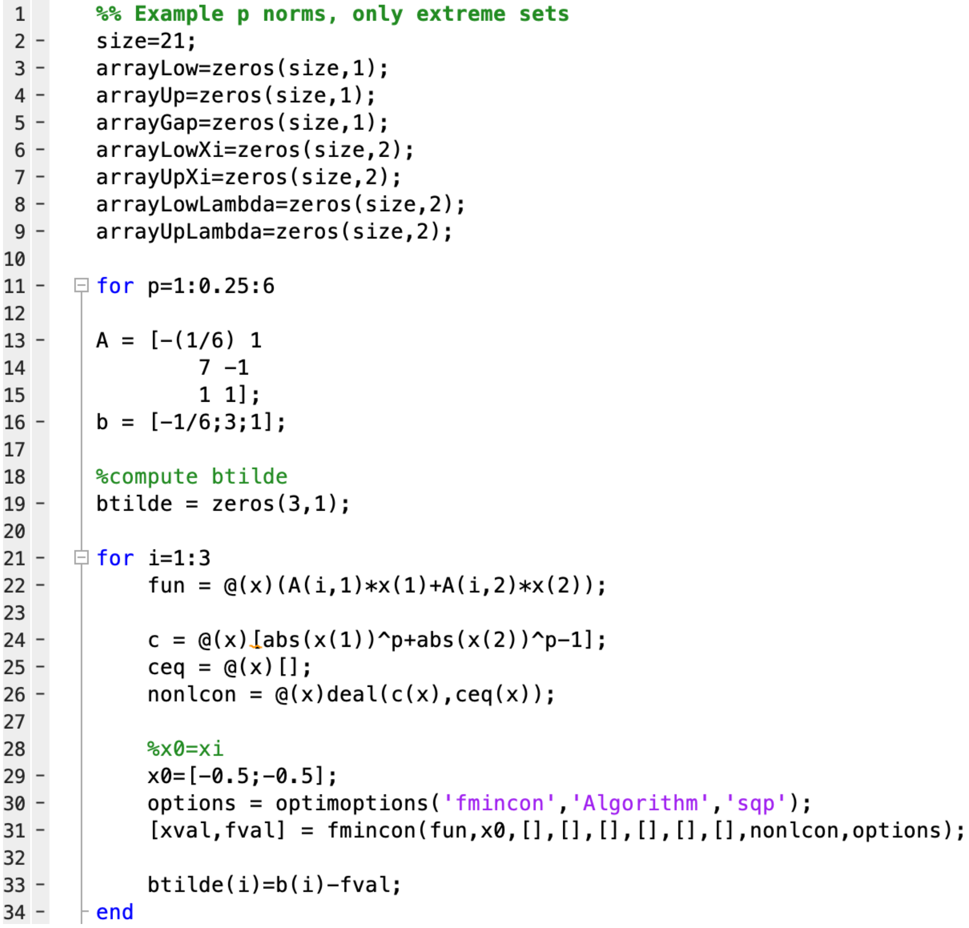
In the further future, it would be interesting to look at convex quadratic problems and find formulations of the bounds in these cases. It might be possible to define new bounds which could either give smaller intervals for the gap, lead to new information for a decision, or which would be easier to compute.

One last idea is to define the robustness gap also for other kinds of optimization such as network optimization.

Appendix

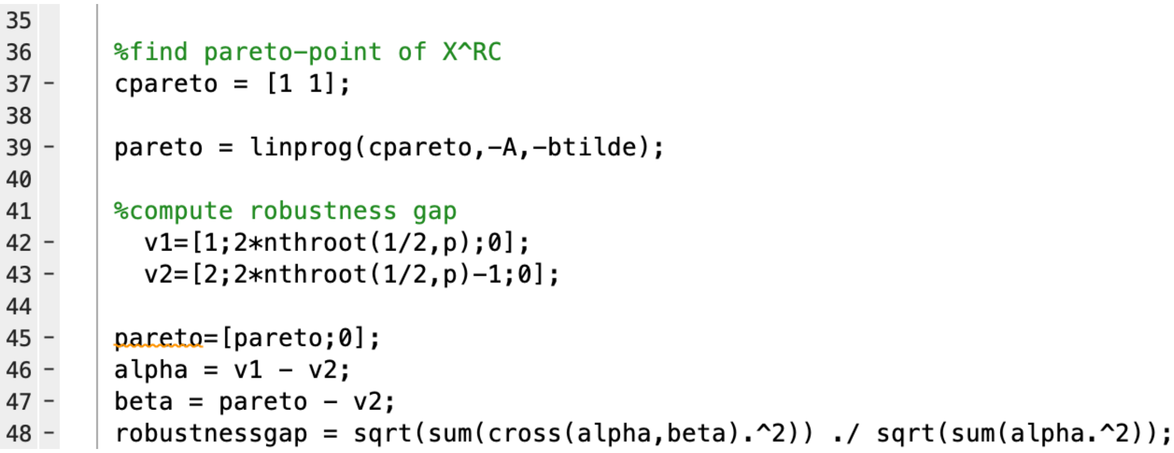
Code

The MATLAB code for solving the uncertain BOLP in Section 4.3 is given. We follow the pseudo-code of Algorithm 1 and 2. We first create arrays to save final results to be able to use them later. We then compute $\tilde{b}$.



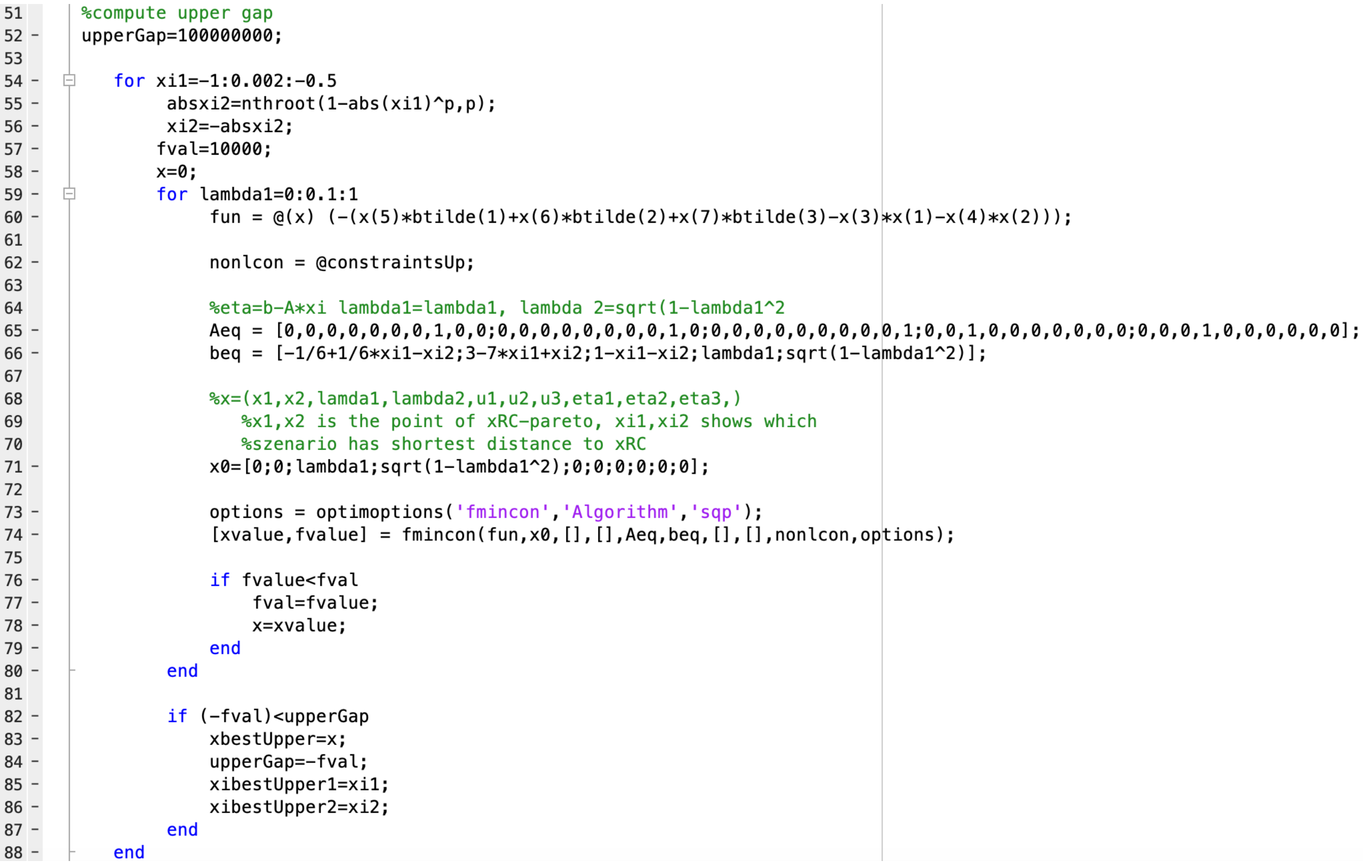
```
%% Example p norms, only extreme sets
size=21;
arrayLow=zeros(size,1);
arrayUp=zeros(size,1);
arrayGap=zeros(size,1);
arrayLowXi=zeros(size,2);
arrayUpXi=zeros(size,2);
arrayLowLambda=zeros(size,2);
arrayUpLambda=zeros(size,2);

for p=1:0.25:6

A = [-(1/6) 1
        7 -1
        1 1];
b = [-1/6;3;1];

%compute btilde
btilde = zeros(3,1);

for i=1:3
    fun = @(x)(A(i,1)*x(1)+A(i,2)*x(2));

    c = @(x)[abs(x(1))^p+abs(x(2))^p-1];
    ceq = @(x)[];
    nonlcon = @(x)deal(c(x),ceq(x));

    %x0=xi
    x0=[-0.5;-0.5];
    options = optimoptions('fmincon','Algorithm','sqp');
    [xval,fval] = fmincon(fun,x0,[],[],[],[],[],[],nonlcon,options);

    btilde(i)=b(i)-fval;
end
```

With this information, we are able to compute the single Pareto point $\{\bar{z}\} = \mathcal{Y}_P^{RC}$ and the robustness gap ϑ.



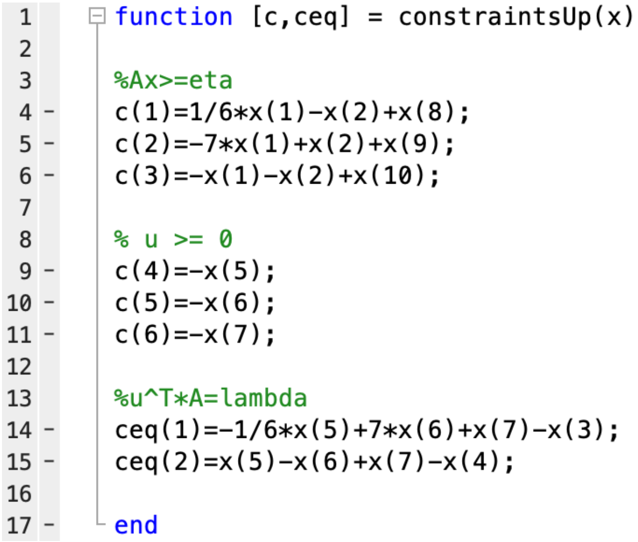
```
%find pareto-point of X^RC
cpareto = [1 1];

pareto = linprog(cpareto,-A,-btilde);

%compute robustness gap
  v1=[1;2*nthroot(1/2,p);0];
  v2=[2;2*nthroot(1/2,p)-1;0];

pareto=[pareto;0];
alpha = v1 - v2;
beta = pareto - v2;
robustnessgap = sqrt(sum(cross(alpha,beta).^2)) ./ sqrt(sum(alpha.^2));
```

We now compute the upper bound Δ^U as described in Algorithm 2 and save the results.



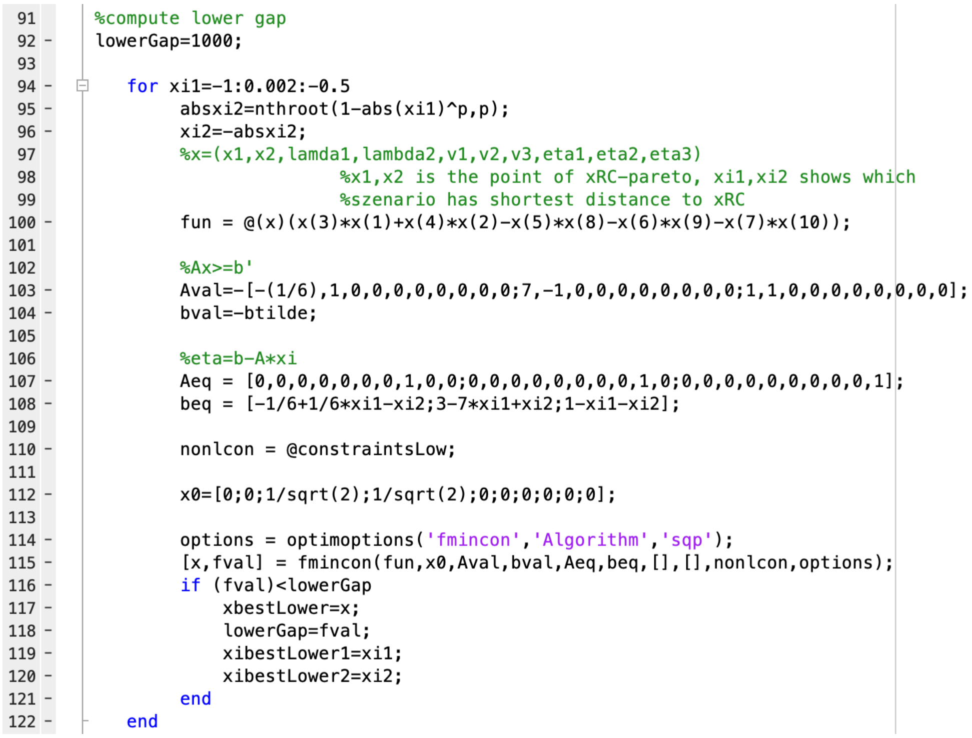
```
%compute upper gap
upperGap=100000000;

   for xi1=-1:0.002:-0.5
        absxi2=nthroot(1-abs(xi1)^p,p);
        xi2=-absxi2;
       fval=10000;
       x=0;
       for lambda1=0:0.1:1
            fun = @(x) (-(x(5)*btilde(1)+x(6)*btilde(2)+x(7)*btilde(3)-x(3)*x(1)-x(4)*x(2)));

            nonlcon = @constraintsUp;

            %eta=b-A*xi lambda1=lambda1, lambda 2=sqrt(1-lambda1^2
            Aeq = [0,0,0,0,0,0,0,1,0,0;0,0,0,0,0,0,0,0,1,0;0,0,0,0,0,0,0,0,0,1;0,0,1,0,0,0,0,0,0,0;0,0,0,1,0,0,0,0,0,0];
            beq = [-1/6+1/6*xi1-xi2;3-7*xi1+xi2;1-xi1-xi2;lambda1;sqrt(1-lambda1^2)];

            %x=(x1,x2,lamda1,lambda2,u1,u2,u3,eta1,eta2,eta3,)
               %x1,x2 is the point of xRC-pareto, xi1,xi2 shows which
               %szenario has shortest distance to xRC
            x0=[0;0;lambda1;sqrt(1-lambda1^2);0;0;0;0;0;0];

            options = optimoptions('fmincon','Algorithm','sqp');
            [xvalue,fvalue] = fmincon(fun,x0,[],[],Aeq,beq,[],[],nonlcon,options);

            if fvalue<fval
                fval=fvalue;
                x=xvalue;
            end
        end

        if (-fval)<upperGap
            xbestUpper=x;
            upperGap=-fval;
            xibestUpper1=xi1;
            xibestUpper2=xi2;
        end
   end
```

The fmincon function uses the nonlinear constraints saved in constraintsUp as showed below.



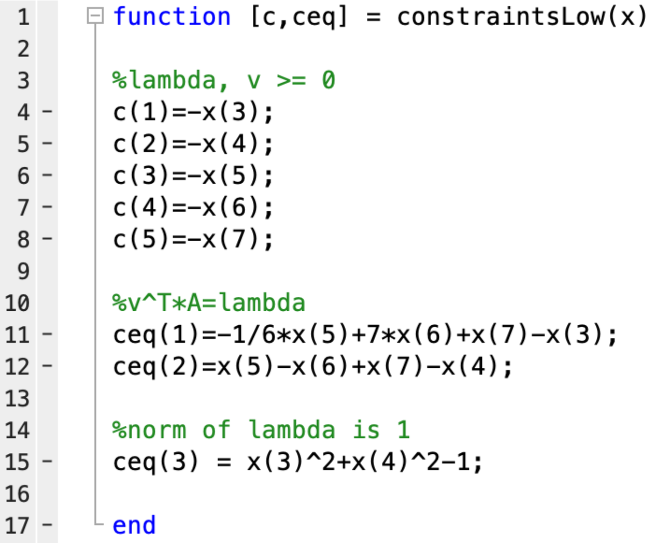
```
function [c,ceq] = constraintsUp(x)

%Ax>=eta
c(1)=1/6*x(1)-x(2)+x(8);
c(2)=-7*x(1)+x(2)+x(9);
c(3)=-x(1)-x(2)+x(10);

% u >= 0
c(4)=-x(5);
c(5)=-x(6);
c(6)=-x(7);

%u^T*A=lambda
ceq(1)=-1/6*x(5)+7*x(6)+x(7)-x(3);
ceq(2)=x(5)-x(6)+x(7)-x(4);

end
```

We now compute the lower bound Δ^L as described in Algorithm 1 and save the results.



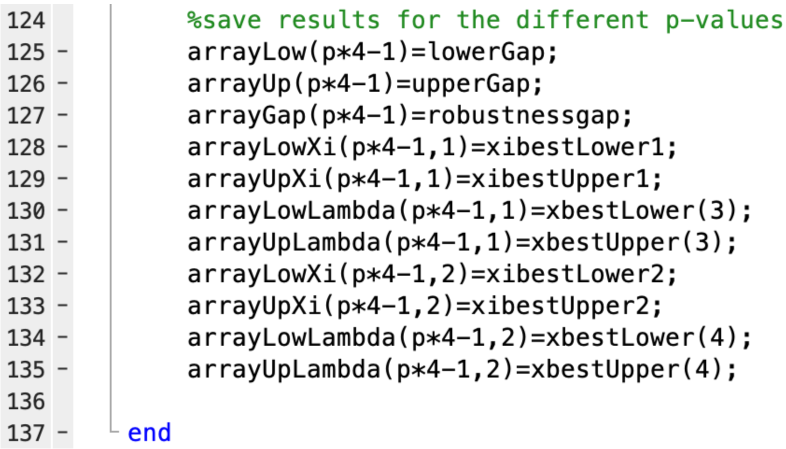
```
%compute lower gap
lowerGap=1000;

    for xi1=-1:0.002:-0.5
          absxi2=nthroot(1-abs(xi1)^p,p);
          xi2=-absxi2;
          %x=(x1,x2,lamda1,lambda2,v1,v2,v3,eta1,eta2,eta3)
                         %x1,x2 is the point of xRC-pareto, xi1,xi2 shows which
                         %szenario has shortest distance to xRC
          fun = @(x)(x(3)*x(1)+x(4)*x(2)-x(5)*x(8)-x(6)*x(9)-x(7)*x(10));

          %Ax>=b'
          Aval=-[-(1/6),1,0,0,0,0,0,0,0,0;7,-1,0,0,0,0,0,0,0,0;1,1,0,0,0,0,0,0,0,0];
          bval=-btilde;

          %eta=b-A*xi
          Aeq = [0,0,0,0,0,0,0,1,0,0;0,0,0,0,0,0,0,0,1,0;0,0,0,0,0,0,0,0,0,1];
          beq = [-1/6+1/6*xi1-xi2;3-7*xi1+xi2;1-xi1-xi2];

          nonlcon = @constraintsLow;

          x0=[0;0;1/sqrt(2);1/sqrt(2);0;0;0;0;0;0];

          options = optimoptions('fmincon','Algorithm','sqp');
          [x,fval] = fmincon(fun,x0,Aval,bval,Aeq,beq,[],[],nonlcon,options);
          if (fval)<lowerGap
              xbestLower=x;
              lowerGap=fval;
              xibestLower1=xi1;
              xibestLower2=xi2;
          end
    end
```

The fmincon function uses the nonlinear constraints saved in constraintsLow as showed below.



```
function [c,ceq] = constraintsLow(x)

%lambda, v >= 0
c(1)=-x(3);
c(2)=-x(4);
c(3)=-x(5);
c(4)=-x(6);
c(5)=-x(7);

%v^T*A=lambda
ceq(1)=-1/6*x(5)+7*x(6)+x(7)-x(3);
ceq(2)=x(5)-x(6)+x(7)-x(4);

%norm of lambda is 1
ceq(3) = x(3)^2+x(4)^2-1;

end
```

At the end we save all computed values to analyze and plot them later.



```
        %save results for the different p-values
        arrayLow(p*4-1)=lowerGap;
        arrayUp(p*4-1)=upperGap;
        arrayGap(p*4-1)=robustnessgap;
        arrayLowXi(p*4-1,1)=xibestLower1;
        arrayUpXi(p*4-1,1)=xibestUpper1;
        arrayLowLambda(p*4-1,1)=xbestLower(3);
        arrayUpLambda(p*4-1,1)=xbestUpper(3);
        arrayLowXi(p*4-1,2)=xibestLower2;
        arrayUpXi(p*4-1,2)=xibestUpper2;
        arrayLowLambda(p*4-1,2)=xbestLower(4);
        arrayUpLambda(p*4-1,2)=xbestUpper(4);

    end
```

Bibliography

[1] A. Ben-Tal and A. Nemirovski. Robust convex optimization. *Mathematics of Operations Research*, 23(4):769–805, 1998.

[2] A. M. Geoffrion. Proper efficiency and the theory of vector maximization. *Journal of Mathematical Analysis and Applications*, 22(3):618 – 630, 1968.

[3] N. T. B. Kim and D. T. Luc. Normal cones to a polyhedral convex set and generating efficient faces in linear multiobjective programming. *Acta Mathematica Vietnamica*, 25(1), 2000.

[4] C. Krüger, A. Schöbel, and M. Wiecek. The robustness gap for uncertain multiobjective optimization. http://hdl.handle.net/11858/00-1735-0000-002E-E3CE-E, 2017, submitted for publication.

[5] D. Kuroiwa and G. M. Lee. On robust multiobjective optimization. *Vietnam Journal of Mathematics*, 40(2&3):305–317, 2012.