A Quick-and-Dirty Approach to Robustness in Linear Optimization

by

Mehdi Karimi

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Combinatorics and Optimization

Waterloo, Ontario, Canada, 2012

Abstract

We introduce methods for dealing with linear programming (LP) problems with uncertain data, using the notion of weighted analytic centers. Our methods are based on high interaction with the decision maker (DM) and try to find solutions which satisfy most of his/her important criteria/goals. Starting with the drawbacks of different methods for dealing with uncertainty in LP, we explain how our methods improve most of them. We prove that, besides many practical advantages, our approach is theoretically as strong as robust optimization. Interactive cutting-plane algorithms are developed for concave and quasi-concave utility functions. We present some probabilistic bounds for feasibility and evaluate our approach by means of computational experiments.

Acknowledgements

I want to thank to all people who helped me with the writing of this Thesis.

First and foremost, I offer my sincerest gratitude to my supervisor, Prof. Levent Tunçel, for his extreme patience, his wise guidance and inspiration, and his tremendous support through out the process of preparing this Thesis. One simply could not wish for a better or friendlier supervisor.

Thanks to Prof. Stephen Vavasis and Prof. Henry Wolkowicz for their valuable comments and careful reading of the draft. Most important of all, I would like to express my deep gratitude to my family and my friends for their generous support and encouragement.

Dedication

I lovingly dedicate this thesis to my wife, Mehrnoosh.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

The idea of dealing with uncertainty by using the notion of weighted analytic center was previously considered in S. Moazeni's Master's thesis [34]. She developed many of the ideas we will use in this thesis. However, in this thesis, we take these ideas further by proving many required results for the weight space that helps us design sophisticated algorithms with many desired properties. We have a thorough probabilistic analysis that is an inherent part of the optimization problems with uncertainty. The main contribution of this thesis is the development of cutting-plane algorithms that make it possible to implement such ideas.

In this chapter, we first discuss different approaches to deal with uncertainty in optimization in Section 1.1. After that, in Section 1.2 we introduce the notations and assumptions we use in this thesis. We introduce some of the required properties for concave functions in Section 1.3. Section 1.4 is the overview of the thesis.

1.1 Uncertainty in optimization

Optimization problems are widespread in real life decision making situations. However, the data perturbations cannot be avoided. In practice, obtaining exact information about the data, at the time when the solution has to be determined, may be difficult or impossible. Therefore, some aspects of the data of the optimization problem are usually uncertain. This uncertainty is caused by many sources such as forecasting or data approximation. As an example, in an agriculture company:

- Factors such as change in government policies or emergence of new products can cause demand uncertainty in the marketing.
- The amount of rain next year is clearly uncertain which can have a profound effect on the agriculture sector among others.

- The amount of profit depends on the prices of the products which is uncertain. This uncertainty is caused by factors such as cost of raw materials, customers budgets and willingness to pay.

In order to handle optimization problems under uncertainty, several techniques have been proposed. The most common approaches are

- Sensitivity analysis,
- Chance constrained programming,
- Stochastic programming,
- Robust optimization.

In the following sections, we will explain the strengths and drawbacks of each of them, going into more details for robust optimization as it is more related to our approach.

1.1.1 Sensitivity analysis

In sensitivity analysis, the influence of the data uncertainty is ignored, and then the obtained solution is justified based on the data perturbations [8, 9]. Sensitivity analysis shows how much the optimal solution to a perturbed problem can differ from the one of the nominal problem. In other words, it gives information about the local stability of a solution without any clue about improving it. This method is also impractical for large number of uncertain parameters.

1.1.2 Chance constrained programming

In chance constrained programming, we use the stochastic models of uncertain data to replace the deterministic constraints by their probabilistic counterparts [27, 11, 20]. It is a natural way of converting the uncertain optimization problem in to a deterministic one. However, most of the time the result is a computationally intractable problem for two reasons [2]; *i)* evaluation of the probabilities with high accuracy is difficult even for simple probability distributions. *ii)* most of the time, the feasible set of the result problem is non-convex which makes the utilization of chance constrained programming highly problematic.

1.1.3 Stochastic programming

In stochastic programming the goal is to find a solution that is feasible for all (or almost all) possible instances of the data and optimizes the expectation of some function of the decisions and the random variables. The most widely used stochastic programming models are two-stage programs. At the first stage, the DM makes a decision. After that, a random event occurs and the solution from the first stage might not satisfy some of the constraints. At the second stage, a recourse decision is made that compensates for any bad effects of the first solution.

The main assumption is that probability distributions governing the data are known or can be estimated which is a major drawback in many applications. Distributions of the random parameters are almost never known exactly, and have to be estimated which yields an approximate solution. Another problem with stochastic programming is that the problems can become unmanageably huge to be able to draw valid conclusions. It is discussed in [35] (for supply chain networks) that the number of scenarios might become huge even for a small problem.

1.1.4 Robust optimization

Robust optimization is the method that is most closely related to our approach. Generally speaking, robust optimization can be applied to any optimization problem where the uncertain data can be separated from the problem's structure. This method is applicable to general convex optimization problem like conic optimization or semidefinite programming [2]. Our concentration in this thesis is on linear programming problems. Uncertainty in the data means that the exact values of the data are not known, at the time when the solution has to be determined. In robust optimization framework, uncertainty in the data is described through *uncertainty sets* that contain all (or most of) possible values that may be realized for the uncertain parameters.

In the last few years, substantial research on robust linear programming with different aspects of uncertainty has been done. In usual robust optimization, one looks for a solution that remains feasible in all cases of the data with a high probability [5, 3, 4, 16, 15, 17, 13]. The first robust optimization approach for LP in the literature was proposed by Soyster [10]. This approach finds a solution that protects against the worst-case scenario remains feasible in all cases of the data. This approach is highly conservative and many problems do not have such solution while still a solution is needed. For instance, when a company is preparing a schedule for a part of a bigger chain, one of the main causes of uncertainty is the completion time of each step. Scheduling according to the worst cases may result in an infeasible problem or a schedule with low benefit, and therefore useless for the company. Therefore, a methodology that is able to give a slightly lower weights to the worst case possibilities can provide a solution to the problem, whereas robust optimization cannot.

Since the interest in robust formulations was revived in the 1990s, many researchers have introduced new formulations for robust optimization framework in linear programming and general convex programming [3, 4, 5, 6, 16, 15, 17, 13, 25]. Ben-Tal and Nemirovski [3, 4] provided some of the first formulations for robust LP with detailed analysis. D. Bertsimas and M. Sim proposed an approach that offers control on the degree of conservatism for every constraint. D. Bertsimas et al. [13] characterize the robust counterpart of an LP problem with uncertainty set described by an arbitrary norm. By choosing appropriate norms, they recover the formulations proposed in the above papers [3, 4, 13].

The goal of classic robust optimization is to find a solution that is capable to cope best of all with *all* realizations of the data from a given (usually bounded) uncertainty set [2, 7]. By the classic definition of robustness [2, 5, 17, 28], the robust optimal solution is the solution of the following problem:

$$\max_{x\in\mathbb{R}^n} \left\{ \inf_{\tilde{c}\in\mathcal{C}} \langle \tilde{c}, x\rangle : Ax \leq \tilde{b} \ , \forall \tilde{b} \in \mathcal{B} \right\}, \tag{1.1}$$

where $\mathcal{C}$ and $\mathcal{B}$ are given uncertainty sets for $\tilde{c}$ and $\tilde{b}$, respectively. Throughout this thesis, we refer to the formulation of (1.1) as *classic robust formulation*. In classic robust optimization, we usually accept the following assumptions [2]:

- All decision variables represent "here and now" decisions, i.e., the values of all the variables should be specified for solving an optimization problem, before the actual data "reveal themselves". This assumption is made in most of the literature such as [15, 5] and is denoted *Non-adjustable robust optimization*. *Adjustable* robust optimization is also considered for example in [6] in which only part of the decisions are "here and now" ones, while the remaining variables represent "wait and see" decisions. Our proposed approach can be regarded as a non-adjustable case.

- We or the DM can specify an appropriate (usually bounded) uncertainty set, and the DM is fully responsible for consequences of his/her decisions when and only when, the actual data is within this set.

- The constraints are " hard" as even small violations of the constraints cannot be tolerated. As we will explain in Chapter 2, our approach does not completely satisfy this assumption.

1.1.5 Drawbacks of robust optimization

Classical robust optimization is a powerful method to deal with optimization problems with uncertain data, however, there are some criticisms of this method as we explain in the following.

As we mentioned, one of the assumptions for robust optimization is that the uncertainty set must be precisely specified before solving the problem. However, in practice, the DM can specify an *estimate* of the borders of uncertainty set. Even if the uncertainty is only in the RHS, expecting the DM to construct accurately an ellipsoid for uncertainty set is not always reasonable.

The main criticism to classical robust optimization is that satisfying all of the constraints, if not make the problem infeasible, may lead to an objective value very far from the optimal value of the nominal problem. This problem is more critical for large deviations.

As an example, [34] considered some of the problems of NETLIB library. Assume that the entries of the right-hand-side (RHS) vector of the inequality constraints in each problem is subject to uncertainty. I.e., $\tilde{b}_i = b_i^{(0)} + \epsilon\xi_i$, where ξ_i is a symmetric random variable in $[-1, 1]$. For each problem, by partitioning the constraints into inequality constraints and equality constraints, we arrive at nominal inequality constraints $A^{(1)}x \leq b^{(1)}$, and equality constraints, $A^{(2)}x = b^{(2)}$. Define

$$\epsilon^\star := \inf\left\{\epsilon : \left\{x \in \mathbb{R}^n : A^{(1)}x \leq b^{(1)} - \epsilon e, \quad A^{(2)}x = b^{(2)}\right\} = \varnothing\right\}. \tag{1.2}$$

The value of $\epsilon^\star$ is shown in Table 1.1 for some of the NETLIB problems. As it is clear from Table 1.1, classic robust counterpart of most of the problems in NETLIB, becomes infeasible for a small perturbation. Moreover, in other problems, objective value of the classic robust optimal solution is very low and unsatisfactory for the decision maker. Examples show that we can find a much better solution in the sense of objective function by reducing the feasibility guarantee of just a few constraints.

Several modifications of classical robust optimization have been introduced to deal with this defect. One, for example, is *Globalized robust counterparts* introduced in Chapter 3 of [2]. The idea is to consider some constraints as "soft" whose violation can be tolerated to some degree. In this method, we take care of what happens when data run out of the nominal uncertainty set. In other words, we have " controlled deterioration" of the constraint. These modified approaches have more flexibility than the classic robust methodology, but we have the problem that the modified robust counterpart of uncertain problems may become computationally intractable.

Although the modified robust optimization frameworks rectify this drawback to some extent, they intensify the first criticism by putting more pressure on the DM to specify deterministic uncertainty sets before solving the problem. It is generally difficult for the DM, for example, to distinguish between soft and hard constraints. Moreover, after solving the robust counterpart, if the DM wants to modify the uncertainty set, in many cases s/he should solve the problem again.

Another criticism of the classic robust optimization is that it gives the same "weight" to all the constraints. In practice, this is not the case as some constraints are more important

Table 1.1: The value of $\epsilon^\star$ defined in (1.2) for some of the NETLIB problems [34].

Problem	ineq	eq	$\epsilon^\star$
E226	190	33	$0(constraint 130)$
BRANDY	54	166	$0(constraint 22, 121)$
FINNIS	450	47	$0(constraint 485, 486, 493)$
BORE3D	19	214	$0(constraint 232)$
SCRS8	106	384	$0(constraint 17)$
SC50A	30	20	$0(constraint 3)$
BLEND	31	43	$(0.4806, 0.4807]$
ADLITTLE	41	15	$(0.37247, 0.37248]$
ISRAEL	174	0	$(30.0912, 30.0913]$
AGG	452	36	$(1.0e-007, 1.0e-006]$
AGG2	456	60	$(0.2322, 0.2323]$
AGG3	456	60	$(0.2544, 0.2545]$
SCAGR7	45	84	$(45.565, 45.566]$
SCAGR25	171	300	$(45.849, 45.850]$

for the DM. There are some options in classical robust optimization like changing the uncertainty set which again intensifies the first criticism. We will see that our approach can completely remove this problem.

1.2 Notations and assumptions

Before introducing our approach in the next chapter, let us first explain some of the assumptions and notations we are going to use. Much of the prior work on robust linear programming addresses the uncertainty through the coefficient matrix. Bertsimas and Sim [15] considered linear programming problems in which all data except the right-hand-side (RHS) vector is uncertain. In [5, 3, 17], it is assumed that the uncertainty affects the coefficient matrix and the RHS vector. Some papers deal with uncertainty only in the coefficient matrix [4, 16, 13]. Optimization problems in which all of the data in the objective function, RHS vector and the coefficient matrix are subject to uncertainty, has been considered in [6]. In this thesis, we concentrate on LP problems for which the coefficient matrix is deterministic, where the coefficients of the objective function and the RHS vector are subject to uncertainty.

As we will explain in Chapter 2, the nominal data and a rough approximation of the uncertainty set are enough for our approach. Hence, we can approximate the uncertainty in the coefficient matrix with the uncertainty in the RHS and the objective function. In other words, we may fix the coefficient matrix on one of the samples from the uncertainty set and then handle the uncertainty by introducing uncertainty to the RHS vector. Besides that, this formulation is typical for many real world problems as we explain below.

In many applications of planning and network design problems such as scheduling, manufacturing, electric utilities, telecommunications, inventory management and transportation, uncertainty only affects costs (coefficients of the objective function) and demands (the RHS vector)[33, 21]:

- **Transportation system:** In many problems in this domain, we can assume that in a road network, the nodes and the arcs are fixed. However, the cost associated to each arc, i.e. the vehicle travel time, and/or the capacity associated to each arc are not known precisely.
- **Traffic assignment problem:** We assume that the drivers have perfect information about the arcs and nodes, which are the structure of the road network and the existing streets. However, their route choice behavior makes the travelling time uncertain.
- **Distribution system:** The place of warehouses and their capacities in inventory planning and distribution problems are well-known and fixed for the DM. However, the number of orders and the demand rate of an item would translate to an uncertain RHS vector. Holding cost, set up cost and shortage cost, which affect the optimal inventory cost, are also flexible and/or uncertain. These affect the objective function

In all of the aforementioned applications, well-understood existing resources, reliable structures (well-established street and road networks, warehouses, machines,..., which are not going to change), and logical components of the formulation are translated into a certain coefficient matrix. The data in the objective function and the RHS vector are usually specified subjectively by the DM or affected by uncertain elements such as institutional, social, or economical ones. Therefore, determining the quantity of these coefficients with precision is often difficult or practically impossible. Hence, considering uncertainty in the objective function and the right-hand-side vector seems to be very applicable, and motivates us to consider such formulation in LP problems separately.

An uncertain linear programming problem with deterministic coefficient matrix $A \in \mathbb{R}^{m\times n}$, is of the form:

$$\begin{aligned} \max \quad & \langle \tilde{c}, x\rangle \\ \text{s.t.} \quad & Ax \leq \tilde{b}, \\ & x \in \mathbb{R}^n, \end{aligned} \tag{1.3}$$

where $\tilde{c} \in \mathcal{C}$ and $\tilde{b} \in \mathcal{B}$ are an n-vector and an m-vector respectively, whose entries are subject to uncertainty. $\mathcal{C}$ and $\mathcal{B}$ are called *uncertainty sets*. In this thesis, we deal with problem (1.3) and suppose that the data uncertainty affects only the elements of the vectors $\tilde{b}$ and $\tilde{c}$. We assume entries of $\tilde{c}$ and $\tilde{b}$ are random variables with unknown distributions, as it is impractical to assume that the exact distribution is explicitly known. By classical view of robust optimization, *classic robust counterpart* of problem (1.3) is defined in (1.1). Feasible/Optimal solutions of problem (1.1) are called *classic robust feasible/classic robust optimal solutions* of problem (1.3) [2].

We usually just have reasonable estimates of the expected value of the uncertain data. In the proposed approach, having nominal (expected) values of the uncertain data is enough to implement. However, in order to compare our algorithm with classical robust ones, and providing probability bounds to the DM, we make the following assumptions on the data:

- For every $i \in \{1, 2, \cdots, m\}$, $\tilde{b}_i$ can be written as $\tilde{b}_i = b_i^{(0)} + \sum_{l=1}^{N_i} \Delta b_i^l \tilde{z}_i^l$ where $\{\tilde{z}_i^l\}_{l=1}^{N_i}$ are independent random variables for every $i \in \{1, \cdots, m\}$.
- For each $\tilde{c}_i$, $i \in \{1, \cdots, n\}$, we have $\tilde{c}_i = c_i^{(0)} + \sum_{l=1}^{N_{ic}} \Delta c_i^l \tilde{z}_{ic}^l$ where $\{\tilde{z}_{ic}^l\}_{l=1}^{N_{ic}}$ are independent random variables.

As can be seen above, each variable $\tilde{b}_i$ is the summation of a nominal value $b_i^{(0)}$ with scaled random variables $\{\tilde{z}_i^l\}_{l=1}^{N_i}$. In practice, the number of these random variables N_i is small compared to the dimension of A as each random variable $\tilde{z}_i^l$ represents a major source of uncertainty in the system. In other words, we can argue that we approximate the uncertainty in A with a handful of random variables in the RHS as in [14].

Here, we impose the following restrictions on the problem (1.3) [34]:

- The matrix A has full column rank, i.e., $\text{rank}(A) = n \leq m$.
- The set $\{x \in \mathbb{R}^n : Ax \leq b^{(0)}\}$ is bounded.
- The set $\{x \in \mathbb{R}^n : Ax \leq b^{(0)}\}$ has nonempty interior.

If A does not have full column rank, then a problem of the form

$$\begin{aligned} \max \quad & \langle c^{(0)}, x \rangle \\ \text{s.t.} \quad & Ax \leq b^{(0)}, \\ & x \in \mathbb{R}^n, \end{aligned} \tag{1.4}$$

either is unbounded, or can be projected to a smaller dimensional space, with the same optimal value. Let μ be a $(n-1)$-vector such that $\sum_{i=1}^{n-1} \mu_i A_i = A_n$, where A_i is the i-th column of A. By some simple linear algebra, we can eliminate x_n and represent problem

(1.3) in the $(n-1)$-dimensional space. If the problem is not unbounded, the optimal objective function value of the problem in the $(n-1)$-dimensional space equals the optimal objective function value of the original problem. If the LP represent a practical problem such as a combinatorial optimization problem, nonnegativity constraints imply that A has a full column rank.

The assumption on the boundedness of $\{x \in \mathbb{R}^n : Ax \leq b^{(0)}\}$ is not very restrictive as it is satisfied in many practical problems representing integer programming and combinatorial optimization problems.

For the third property, assume that the polyhedron $\mathcal{P} := \{x \in \mathbb{R}^n : Ax \leq b^{(0)}\}$ is nonempty with empty interior. This means that the dimension of the affine hull of $\mathcal{P}$, say d, is less than n. Hence, we can represent the affine space of $\mathcal{P}$ as $\{x = h + By : y \in \mathbb{R}^d\}$, where B has full column rank d. We can define a polyhedron $\hat{\mathcal{P}} := \{x \in \mathbb{R}^d : \hat{A}x \leq \hat{b}\}$ with non-empty interior such that there is a one-to-one map between $\mathcal{P}$ and $\hat{\mathcal{P}}$, using the fact that B has full column rank. We can rewrite the problem in $\mathbb{R}^d$ with feasible region $\hat{\mathcal{P}}$.

In this thesis, vectors and matrices are denoted, respectively, by lower and uppercase letters. The matrices Y and S represent diagonal matrices, having the components of vectors y and s on their main diagonals, respectively. The letter e and e_i denote a vector of ones and a vector that is everywhere zero except at the i-th entry with the appropriate dimension, respectively. The rows of a matrix are shown by superscripts of the row, i.e., $a^{(i)}$ is the i-th row of the matrix A. The inner product of two vectors $a, b \in \mathbb{R}^n$ is shown both by $\langle a, b \rangle$ and $a^T b$. For a matrix A, we show the range of A with $\mathcal{R}(A)$ and the null space of A with $\mathcal{N}(A)$.

1.3 Convex functions and subgradient

As we are going to design algorithms for concave utility functions, we express the required results for concave functions. These results are generally proven for the convex functions in the textbooks [31, 32], however we can translate all of them to concave functions.

A function $f : \mathbb{R}^n \to \mathbb{R}$ is convex if the domain of f, $(\text{dom} f)$, is convex and for any $x, y \in \text{dom} f$, and $\alpha \in [0, 1]$ we have

$$f(\alpha x + (1-\alpha)y) \leq \alpha f(x) + (1-\alpha) f(y). \tag{1.5}$$

A function is concave if $-f$ is convex. We say that the function is *strictly* convex if strict inequality holds in (1.5), and similarly *strictly* concave if $-f$ is strictly convex. Usually, we extend the definition of f over $\mathbb{R}^n$ by giving the value of ∞ to all $x \notin \text{dom} f$. An example of a both concave and convex function is *affine function* $f(x) = a^T x + b$, $a \in \mathbb{R}^n$ and $b \in \mathbb{R}$. An example of a concave function is $f(x) = \sum_{i=1}^n t_i \ln(x_i)$ where $t \in \mathbb{R}^n_+$, and $\text{dom} f = \mathbb{R}^n_{++}$.

Concave and convex functions have many properties that make them interesting for optimization plus the fact that many practical problems can be modeled by using them.

Theorem 1.3.1. *For a concave function $f : \mathbb{R}^n \to \mathbb{R}$, any local maximizer is also a global maximizer. If a strictly concave function attains its global maximizer, it is unique.*

The following theorem is very useful for developing our cutting-plane algorithm.

Theorem 1.3.2. *Assume that $f : \mathbb{R}^n \to \mathbb{R}$ is a concave function and let $x^0 \in \operatorname{relint}(\operatorname{dom} f)$. Then there exists $g \in \mathbb{R}^n$ such that*

$$f(x) \leq f(x^0) + g^T(x - x^0), \quad \forall x \in \mathbb{R}^n. \tag{1.6}$$

If f is differentiable at x^0, then g is unique, and $g = \nabla f(x^0)$.

The vector g that satisfies (1.6) is called the *supergradient* of f at x^0. The set of all supergradients of f at x_0 is called the *superdifferential* of f at x^0, and is denoted $\partial f(x^0)$. By Theorem 1.3.2, if f is differentiable at x^0, then $\partial f(x^0) = \{\nabla f(x^0)\}$.

Example 1.3.1. *Consider a one variable function $f(x) = -|x|$. For $x < 0$ we have $\partial f(x) = \{\nabla f(x)\} = \{1\}$. Similarly, for $x > 0$ we have $\partial f(x) = \{-1\}$. It is also easy to show $\partial f(0) = [-1, 1]$.*

The following lemma about supergradient is also useful.

Lemma 1.3.1. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a concave function, and $D \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$ be arbitrary matrices. Then, $g(x) := f(Dx + b)$ is a concave function and we have:*

$$\partial g(x) = D^T \partial f(Dx + b)$$

1.4 Overview of the thesis

In this thesis, we design new algorithms which alleviate some of the drawbacks of classical robust optimization approach mentioned above. We employ an interactive decision making approach to involve DM in the optimization process, and to increase the reliability of the information extracted from the DM. We also utilize the notion of weighted analytic centers, and implement our algorithms in the space of weight vectors which makes the interaction with the DM easier.

In Chapter 2, we explain our approach and the scheme of our algorithm. In Chapter 3, we consider the properties of the weight space that help us to design the algorithm and perform the probabilistic analysis. In Chapter 4, we prove theoretically that our approach is as least as strong as the classical robust optimization approach. Chapter 5 is about the probabilistic analysis that is important for interaction with the decision maker. In Chapter 6, we design the cutting-plane algorithms, talk about the modifications of the algorithm, and explain some practical concerns of our approach. Some preliminary computational results are presented in Chapter 7. In Chapter 8, we briefly talk about the extension of the approach to semidefinite programming and quasi-concave utility functions, and then conclude the thesis.

Chapter 2

Utility function approach

In Chapter 1, we introduced different methods for dealing with LP problems under uncertainty. For each method, we explained the drawbacks and practical difficulties. In this chapter, we want to introduce our new approach that helps us overcome some of these difficulties. Let us focus on the robust optimization method that from many points of view is the strongest among the methods we introduced in Chapter 1. One of the main problems with robust optimization is that the uncertainty region must be specified before solving the problem. As we explained, in practice, even if the uncertainty is only in the RHS, expecting the DM to construct accurately an ellipsoid for uncertainty set is not reasonable.

The proposed method removes DM's anxiety about determining the uncertainty set precisely, and a nominal value of the data is enough. We just need to have an estimate from the uncertainty set to evaluate the proposed solution and derive the probability bounds for our approach. We propose a tractable method to find a solution in the expected feasible region (nominal feasible region) which satisfies the expectations of the decision maker. Although in this approach the robustness of some constraints is ignored, the proposed solution is robust from DM's point of view. In this thesis, we assume that A, $c^{(0)}$, and $b^{(0)}$ are the nominal values of the uncertain LP problem in (1.3). Hence, the nominal LP problem that we use to design our algorithm is (1.4).

The proposed solution is obtained efficiently by using the notion of weighted analytic centers. As we will explain in Chapter 3, there is a correspondence between the feasible region and the weight space. To any weight vector $w \in \mathbb{R}^m_{++}$, we can assign three vectors $(x(w), s(w), y(w))$, $s > 0$, $y > 0$, where $s = b - Ax$ is the slack vector. We will mention that for any feasible vector $\hat{x}$, there exists $w \in \mathbb{R}^m_{++}$ such that $x(w) = \hat{x}$. This property shows that we can sweep the whole feasible region by moving in the weight space. Working in the weight space is equivalent to working with the slack variables which gives a tangible understanding about how far we are from the boundary of the feasible region. As will be explained latter, we can also add a constraint to the problem for the objective function and translate the objective value to a slack variable. This helps us work just with the slack variables to solve the problem.

The conceptual algorithm exploits DM's information (preferences) by repeated (sequencing) decisions (evaluations) over the performance of the solution. Some of the simple questions that can be asked of the DM are pairwise comparison questions (what is the relative importance of constraint i compared to constraint j), objective function is included. Then we can use these to construct the initial weights. We compute $x(w)$ and probabilities of feasibility if some partial distribution information is available and go back to the DM. DM either accepts $x(w)$ and the probabilities, or asks to trade some of the current robustness with respect to constraints I_1 for more robustness for constraints in I_2, where I_1 and I_2 are two groups of constraints.

The above interaction with the DM is not completely clear. We need a more tangible interaction with the DM to design an algorithm. The primary questions we have to answer are:

- How can the DM's preferences be represented as a weight vector?
- Which questions can be asked from the DM?
- How can we move in the slack or weight space based on the answers from the DM?

In this thesis, we assume that the DM's preferences can be modeled by a utility function $U : \mathbb{R}^{(m+1)} \to \mathbb{R}$ (we have m slack variables for m constraints and one slack variable for the objective value). By this assumption, we can write our problem as

$$\begin{array}{rl} \max & U(s) \\ s.t. & s \in B_s, \end{array} \tag{2.1}$$

where B_s is the set of *centric* s-vectors that we define later in Chapter 3 (Definition 3.2.1). We do not have access to this utility function, however assume that, for a centric slack vector s, we can ask the DM for some information about the function. The questions we are going to ask are the supergradient of $U(s)$ at some points and some pairwise comparison questions if needed. The goal of our algorithm is to maximize this utility function. At each step, we use the information from the DM to produce a cut in the s-space or w-space to shrink the corresponding set such that an optimal solution is kept in the shrunken set. The designing of the algorithms, the material about convergence, and practical issues will completely be covered in Chapter 6.

Chapter 3

Weighted analytic centers

In this chapter, we first define the notion of weighted analytic center in Section 3.1. In Section 3.2, we show that we can represent the robust feasible region with the weight vectors. The results of this section are useful for probabilistic analysis in Chapter 5. In Section 3.3, we prove many useful results about the properties of the weight space, which are really useful for the design of the algorithms in Chapter 6.

3.1 Definition of weighted center

For every $i \in \{1, 2, \cdots, m\}$, let $\mathcal{F}_i$ be a closed convex subset of $\mathbb{R}^n$ such that $\mathcal{F} := \bigcap_{i=1}^m \mathcal{F}_i$ is bounded and has nonempty interior.
Let $F_i : \operatorname{int}(\mathcal{F}_i) \to \mathbb{R}$ be a self-concordant barrier for $\mathcal{F}_i$, $i \in \{1, 2, \cdots, m\}$ (For a definition of self-concordant barrier functions see [39]). For every $w \in \mathbb{R}^m_{++}$, we define the w-center of $\mathcal{F}$ as

$$\arg\min \left\{ \sum_{i=1}^m w_i F_i(x) : x \in \mathcal{F} \right\}.$$

Consider the special case when each $\mathcal{F}_i$ is a half-space in $\mathbb{R}^n$. Then the following result is well-known.

Theorem 3.1.1. *Suppose for every $i \in \{1, 2, \cdots, m\}$, $a^{(i)} \in \mathbb{R}^n \setminus \{0\}$ and $b_i \in \mathbb{R}$ are given such that:*

$$\mathcal{F} := \left\{ x \in \mathbb{R}^n : \langle a^{(i)}, x \rangle \le b_i^{(0)}, \forall i \in \{1, 2, .., m\} \right\},$$

is bounded and $\operatorname{int}(\mathcal{F})$ is nonempty. Also, for every $i \in \{1, 2, \cdots, m\}$ define $F_i(x) := -\ln(b_i^{(0)} - \langle a^{(i)}, x \rangle)$. Then for every $w \in \mathbb{R}^m_{++}$, there exists a unique w-center in the interior of $\mathcal{F}$, $x(w)$. Conversely, for every $x \in \operatorname{int}(\mathcal{F})$, there exists some weight vector $w(x) \in \mathbb{R}^m_{++}$ such that x is the unique $w(x)$-center of $\mathcal{F}$.

Define the following convex optimization problems:

$$\begin{aligned} \min \quad & \langle c, x\rangle - \sum_{i=1}^{m} w_i \ln(s_i) \\ \text{s.t.} \quad & Ax + s = b, \\ & s \in \mathbb{R}^m_{++},\ x \in \mathbb{R}^n, \end{aligned} \tag{3.1}$$

and

$$\begin{aligned} \min \quad & \langle b, y\rangle - \sum_{i=1}^{m} w_i \ln(y_i) \\ \text{s.t.} \quad & A^T y = c, \\ & y \in \mathbb{R}^m_{++}, \end{aligned} \tag{3.2}$$

where $A \in \mathbb{R}^{m\times n}$, $b \in \mathbb{R}^m$, and $c \in \mathbb{R}^n$. For every weight vector $w > 0$, the objective functions of the above problems are strictly convex on their domains. Moreover, the objective function values tend to $+\infty$ along any sequence of their interior points (strictly feasible points), converging to a/some point on their respective boundary. So the above problems have minimizers in the interior of their respective feasible regions. Since the objective functions are strictly convex, the minimizers are unique. Therefore, for every given $w > 0$, the above problems have unique solutions $(x(w), s(w))$ and $y(w)$. These solutions can be used to define many *primal-dual weighted-central-paths* as the solution set $\{(x(tw), y(tw), s(tw)) : t > 0\}$ of the following system of equations and strict inequalities:

$$\begin{aligned} & Ax + s = b,\ s > 0, \\ & A^T y = c, \\ & Sy = w, \end{aligned} \tag{3.3}$$

where $S := \mathrm{Diag}(s)$. When we set $w := e$, we obtain the usual primal-dual weighted-central-path. Fig 3.1 shows the primal-dual central paths for a typical feasible region. As we explain later, in this thesis, we add the objective value as a constraint to the problem. Hence, in (3.3) we put $c = 0$, and solve the following set of equations to find the weighted center:

$$\begin{aligned} & Ax + s = b,\ s > 0, \\ & A^T y = 0, \\ & Sy = w, \end{aligned} \tag{3.4}$$

For every given weight vector w, $(x(w), y(w), s(w))$ is obtained uniquely and $x(w)$ is called the *weighted center* of w. We may also refer to $(x(w), y(w), s(w))$ as the weighted center of w. For every given $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$, $y > 0$, that satisfy the above system, w and $s(w)$ are obtained uniquely. However, for a given $x \in \mathbb{R}^n$, there are many weight vectors w that give x as the w-center of the corresponding polytope.

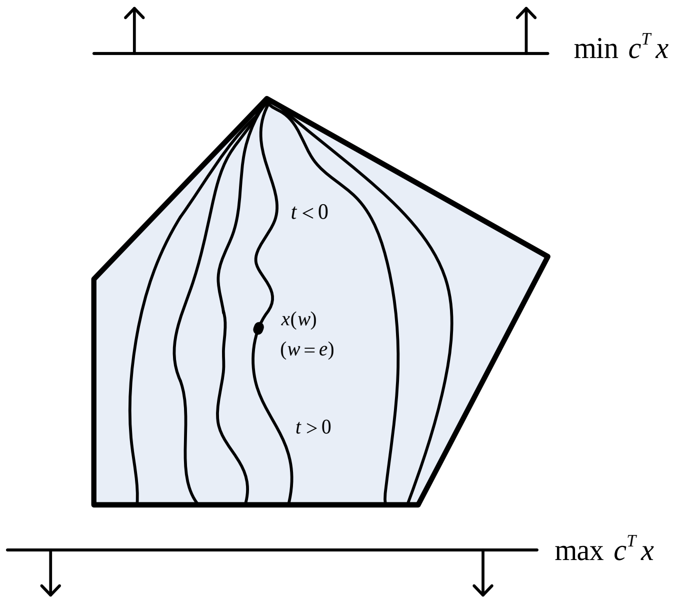

Figure 3.1: Primal-dual central paths.

Example 3.1.1. [34] *Let*

$$b := \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \end{pmatrix} \qquad A := \begin{pmatrix} 1 & 0 \\ -1 & 0 \\ 0 & 1 \\ 0 & -1 \end{pmatrix},$$

then the point $x = (0.5\ \ 0.5)^T$ *is both* $(0.25,\ 0.25,\ 0.25,\ 0.25)$*-center* (*corresponding to* $y = 0.5e$) *and* $(0.35,\ 0.35,\ 0.15,\ 0.15)$*-center* (*corresponding to* $y = (0.7,\ 0.7,\ 0.3,\ 0.3)^T$) *of the polytope.*

The following well-known lemma is really useful.

Lemma 3.1.1. *Let* (x, y, s) *and* $(\hat{x}, \hat{y}, \hat{s})$ *be the solutions of system* (3.4) *corresponding to the weight vectors* w, $\hat{w} \in \mathbb{R}^m_{++}$, *respectively. For every* $\bar{y}$ *in the null space of* A^T *we have:*

$$\langle \hat{s}, \bar{y} \rangle = \langle s, \bar{y} \rangle.$$

Proof. From (3.4), we have $s = b^{(0)} - Ax$ and $\hat{s} = b^{(0)} - A\hat{x}$, which results in $s - \hat{s} = A(x - \hat{x})$. Hence we have $s - \hat{s} \in \mathcal{R}(A)$. As the null space of A^T and the range of A are orthogonal, for every $\bar{y} \in \mathcal{N}(A^T)$ we can write:

$$\langle s - \hat{s}, \bar{y} \rangle = 0 \quad \Rightarrow \quad \langle \hat{s}, \bar{y} \rangle = \langle s, \bar{y} \rangle.$$

□

Let $(\hat{x}, \hat{y}, \hat{s})$ be the solution of system (3.4) corresponding to the weight vector $\hat{w}$. Moreover, assume that $\bar{y} > 0$ is such that $A^T\bar{y} = 0$. Then, by using Lemma 3.1.1, we can show that $(\hat{x}, \bar{y}, \hat{s})$ is the solution of system (3.4) corresponding to the weight vector $\left(\frac{\bar{y}}{\hat{y}}\hat{w}\right)$. Hence, there may be many weight vectors that give the same w-center. In the following lemma, we find an upper-bound and a lower-bound for $s(w)$ for some weight vector $w > 0$.

Lemma 3.1.2. [34] *Let (x, y, s) and $(\hat{x}, \hat{y}, \hat{s})$ be the solutions of system (3.4) corresponding to the weight vectors w, $\hat{w} \in \mathbb{R}^m_{++}$, respectively. Then, for every $i \in \{1, 2, \cdots, m\}$, we have*

$$\left(\frac{\hat{w}_i}{\langle e, \hat{w}\rangle}\right) s_i \leq \hat{s}_i \leq \left(\frac{\langle e, w\rangle}{w_i}\right) s_i.$$

Proof. Since x and $\hat{x}$ are weighted centers of $\mathcal{F}$, we have $Ax + s = b^{(0)}$ and $A\hat{x} + \hat{s} = b^{(0)}$. Therefore, $s - \hat{s} \in \mathcal{R}(A)$. Moreover, $A^Ty = 0$ and $A^T\hat{y} = 0$ imply that $y, \hat{y} \in \mathcal{N}(A^T)$. Using $Sy = w$, we arrive at

$$\langle e, w\rangle = \langle s, y\rangle = \langle s, y\rangle - \langle s - \hat{s}, y\rangle = \langle \hat{s}, y\rangle,$$

where the second equality comes from the fact that $(s - \hat{s}) \in \mathcal{R}(A)$ and $y \in \mathcal{N}(A^T)$.

Nonnegativity of y and $\hat{s}$ imply that $y_i\hat{s}_i \leq \langle \hat{s}, y\rangle = \langle e, w\rangle$. Applying $s_iy_i = w_i$, we conclude

$$\hat{s}_i \leq \left(\frac{\langle e, w\rangle}{w_i}\right) s_i.$$

For the other inequality, similarly, by applying the fact that $\hat{s} - s \in \mathcal{R}(A)$ and $\hat{y} \in \mathcal{N}(A^T)$, we obtain

$$\langle e, \hat{w}\rangle = \langle \hat{s}, \hat{y}\rangle = \langle \hat{s}, \hat{y}\rangle - \langle \hat{s} - s, \hat{y}\rangle = \langle s, \hat{y}\rangle.$$

Thus $s_i\hat{y}_i \leq \langle s, \hat{y}\rangle$ or equivalently $s_i\hat{y}_i \leq \langle e, \hat{w}\rangle$. By using $\hat{y}_i\hat{s}_i = \hat{w}$, we have

$$s_i\frac{\hat{w}_i}{\langle e, \hat{w}\rangle} \leq \hat{s}_i.$$

□

We want to find a relationship between the space of weight vectors and the robust feasible region. We can use Lemma 3.1.2 to derive some probability bounds. However, in some cases this lemma is not tight enough. So, we use Lemma 3.1.1 to find tighter bounds.

3.2 Representing the robust feasible region with weight vectors

In this section, want to relate the notion of weights to the parameters of the uncertainty set. The results of this section are useful for probabilistic analysis in Chapter 5. We consider weighted center (x, y, s) corresponding to weight vectors w such that $\sum_{i=1}^m w_i = 1$. A special case can be $w = \frac{1}{m}e$, where e is the vector of all ones. We will show that this subset of weight vectors is enough to represent the feasible region. We call this simplex of weight vectors W, so

$$W := \{w : w > 0,\ e^T w = 1\}.$$

We can define the following notion for future reference:

Definition 3.2.1. *A vector $s \in \mathbb{R}^m$ or $y \in \mathbb{R}^m$ is called* ***centric*** *if there exists x such that (x, y, s) satisfies (3.4) for a weight vector $w > 0$ where $e^T w = 1$.*

As we explained in Section 1.2, we consider our uncertainty sets as follows:

$$B_i := \left\{ \tilde{b}_i \ : \ \exists \tilde{z} = (\tilde{z}_i^1, \cdots, \tilde{z}_i^{N_i}) \in [-1,1]^{N_i} \ s.t.\ \tilde{b}_i = b_i^{(0)} + \sum_{l=1}^{N_i} \Delta b_i^l \tilde{z}_i^l \right\}, \tag{3.5}$$

where $\{\tilde{z}_i^l\}_{l=1}^{N_i}$, $i \in \{1, \cdots, m\}$ are independent random variables, and Δb_i^l is the scaling factor of $\tilde{z}_i^l$. We assume that the support of $\tilde{z}_i^l$ contains $\tilde{z}_i^l = -1$, i.e., $Pr\{\tilde{z}_i^l = -1\} \neq 0$. Let us define another set which is related to the weight vectors:

$$\mathcal{W} := \left\{ (w_1, \cdots, w_m) : w_i \in [y_i(w)\|\Delta b_i\|_1, 1),\ \sum_{i=1}^m w_i = 1 \right\}, \tag{3.6}$$

where $y(w)$ is the y-vector of w. Our goal is to explicitly specify a set of weights whose corresponding w-center makes the feasible solution of the robust counterpart.

Proposition 3.2.1. *Let x satisfy $Ax \leq \tilde{b}$ for every $\tilde{b} \in B_1 \times \cdots \times B_m$. Then there exists some $w \in \mathcal{W}$, so that x is the weighted analytic center with respect to the weight vector w, i.e., $x = x(w)$. In other words,*

$$\left\{ x \ : \ Ax \leq \tilde{b},\ \forall \tilde{b} \in B_1 \times \cdots \times B_m \right\} \subseteq \{x(w) \ : \ w \in \mathcal{W}\}.$$

Proof. Let $\hat{w} > 0$ be an arbitrary vector such that $\sum_{i=1}^m \hat{w}_i = 1$, and let $(\hat{x}, \hat{y}, \hat{s})$ be the weighted center corresponding to it. Assume that x is in the robust feasible region; we must have $\langle a_i, x\rangle \leq b_i^{(0)} + \langle \Delta b_i, \tilde{z}_i\rangle$ for every $\tilde{z}_i$ with nonzero probability, particularly for $\tilde{z}_i = -e$ where e is all ones vector. So

$$\langle a_i, x\rangle - b_i^{(0)} \leq \langle \Delta b_i, \tilde{z}_i\rangle = \langle \Delta b_i, -e\rangle = -\|\Delta b_i\|.$$

Define $s_i := b_i^{(0)} - \langle a_i, x \rangle$. Thus, from the above equation, for every $i \in \{1, \cdots, m\}$ we have

$$0 < \|\Delta b_i\|_1 \leq s_i,$$

and consequently $\hat{y}_i \|\Delta b_i\|_1 \leq \hat{y}_i s_i$ using the fact that $\hat{y}_i > 0$. For every $i \in \{1, \cdots, m\}$, we set

$$w_i := \hat{y}_i s_i.$$

Since $(x, \hat{y}, s)$ satisfies the optimality conditions, we have $x = x(w)$. It remains to show that $w \in \mathcal{W}$. First note that:

$$\sum_{i=1}^m w_i = \sum_{i=1}^m s_i \hat{y}_i = \sum_{i=1}^m \hat{s}_i \hat{y}_i = \sum_{i=1}^m \hat{w}_i = 1,$$

where for the second equality we used Lemma 3.1.1. Now, using the fact that $w_i \geq 0$ for every $i \in \{1, \cdots, m\}$, we have $w_i < \sum_{j=1}^m w_j = 1$. We already proved that $\hat{y}_i \|\Delta b_i\|_1 \leq \hat{y}_i s_i = w_i$. These two inequalities prove that $w_i \in [\hat{y}_i \|\Delta b_i\|_1, 1)$. □

The above proposition shows that when the robust counterpart problem with respect to the uncertainty set $B_1 \times \cdots \times B_m$ is feasible, the set $\mathcal{W}$ is nonempty. In the next proposition we prove that the equality holds in the above inclusion.

Proposition 3.2.2. *(a)We have*

$$\{x \ : \ Ax \leq \tilde{b}, \ \forall \tilde{b} \in B_1 \times \cdots \times B_m\} = \{x(w) \ : \ w \in \mathcal{W}\}.$$

(b) Assume that $w > 0$ satisfies $\sum_{i=1}^m w_i = 1$, and y is its corresponding y-vector. For every $i \in \{1, \cdots, m\}$, we have

$$w_i \geq y_i \|\Delta b_i\|_1 \Rightarrow \langle a_i, x(w) \rangle \leq \tilde{b}_i, \quad \forall \tilde{b}_i \in B_i.$$

Proof. (a) $\subseteq$ part was proved in Proposition 3.2.1. For $\supseteq$, let $w \in \mathcal{W}$ and (x, y, s) be its corresponding weighted center. By $w \in \mathcal{W}$ we have

$$y_i \|\Delta b_i\|_1 \leq w_i = s_i y_i = (b_i^{(0)} - \langle a_i, x \rangle) y_i \Longrightarrow \|\Delta b_i\|_1 \leq (b_i^{(0)} - \langle a_i, x \rangle).$$

Therefore, for all $\tilde{z}_i \in [-1, 1]^m$,

$$\langle a_i, x \rangle \leq b_i^{(0)} - \|\Delta b_i\|_1 \leq b_i^{(0)} - \sum_{l=1}^N \Delta b_i^l \ \tilde{z}_i^l = b_i^{(0)} + \langle \tilde{z}_i, \Delta b_i \rangle,$$

which proves x is a robust feasible solution with respect to the uncertainty set $B_1 \times \cdots \times B_m$.
(b) Assume that $w > 0$ satisfies $\sum_{i=1}^m w_i = 1$, y is its corresponding y-vector, and there

exists $i \in \{1, \cdots, m\}$ such that $w_i \geq y_i \|\Delta b_i\|_1$. If there exists $\tilde{b}_i \in B_i$ such that $\langle a_i, x(w)\rangle > \tilde{b}_i$ where $\tilde{b}_i = b_i^{(0)} + \sum_{l=1}^{N_i} \Delta b_i^l \tilde{z}_i^l$, by using $\tilde{z}_i^l \geq -1$ we have

$$\begin{aligned}
\langle a_i, x(w)\rangle > \tilde{b}_i \quad &\Rightarrow \quad \langle a_i, x(w)\rangle > b_i^{(0)} + \sum_{l=1}^{N_i} \Delta b_i^l \ \tilde{z}_i^l \ \geq \ b_i^{(0)} - \sum_{l=1}^{N_i} \Delta b_i^l \\
&\Rightarrow \quad \sum_{l=1}^{N_i} \Delta b_i^l > b_i^{(0)} - \langle a_i, x(w)\rangle = s_i(w) \\
&\Rightarrow \quad y_i \sum_{l=1}^{N_i} \Delta b_i^l > y_i s_i(w) = w_i \geq y_i \sum_{l=1}^{N_i} \Delta b_i^l \\
&\Rightarrow \quad \sum_{l=1}^{N_i} \Delta b_i^l \ > \ \sum_{l=1}^{N_i} \Delta b_i^l,
\end{aligned}$$

which is a contradiction. We conclude that $\langle a_i, x(w)\rangle \leq \tilde{b}_i$ for all $\tilde{b}_i \in B_i$. □

We will talk about probability bounds in Chapter 5. Before that, we need to develop further properties of weighted centers. We start by proving the following useful lemma which shows that in some cases, we can find the weighted center for a combination of weight vectors by using the combination of their weighted centers.

Lemma 3.2.1. *Let $(x^{(i)}, y^{(i)}, s^{(i)})$, $i \in \{1, \cdots, l\}$, be solutions of system (3.4), corresponding to the weights $w^{(i)}$. Then for every set of $\beta_i \in [0,1]$, $i \in \{1, \cdots, l\}$, such that $\sum_{i=1}^{l} \beta_i = 1$, and for every $j \in \{1, \cdots, l\}$, we have $(\sum_{i=1}^{l} \beta_i x^{(i)}, y^{(j)}, \sum_{i=1}^{l} \beta_i s^{(i)})$ is the w-center of $\mathcal{F}$, where $w := \sum_{i=1}^{l} \beta_i Y^{(j)} (Y^{(i)})^{-1} w^{(i)}$.*
We also have

$$\sum_{i=1}^{m} w_i = \sum_{i=1}^{m} w_i^{(j)}.$$

Proof. According to the assumptions, for every $i \in \{1, \cdots, l\}$, we have

$$\begin{aligned}
&Ax^{(i)} + s^{(i)} = b^{(0)}, \quad s > 0, \\
&A^T y^{(i)} = 0, \\
&S^{(i)} y^{(i)} = w^{(i)}.
\end{aligned}$$

Now, it can be seen that $(\sum_{i=1}^{l} \beta_i x^{(i)}, y^{(j)}, \sum_{i=1}^{l} \beta_i s^{(i)})$ satisfies the system:

$$\begin{aligned}
&A(\sum_{i=1}^{l} \beta_i x^{(i)}) + (\sum_{i=1}^{l} \beta_i s^{(i)}) = b^{(0)}, \quad (\sum_{i=1}^{l} \beta_i s^{(i)}) > 0, \\
&A^T y^{(j)} = 0, \\
&(\sum_{i=1}^{l} \beta_i S^{(i)}) y^{(j)} = \sum_{i=1}^{l} \beta_i Y^{(j)} (Y^{(i)})^{-1} w^{(i)}.
\end{aligned} \tag{3.7}$$

Since the w-center of $\mathcal{F}$ is unique, the proof for the first part is done.
For the second part, from (3.7) we can write

$$\sum_{i=1}^{m} w_i = \sum_{i=1}^{m}(\sum_{p=1}^{l} \beta_p s_i^{(p)})y_i^{(j)} = \sum_{p=1}^{l} \beta_p(\sum_{i=1}^{m} s_i^{(p)} = \sum_{p=1}^{l} \beta_p \langle s^{(p)}, y^{(j)}\rangle.$$

By Lemma 3.1.1, we have $\langle s^{(p)}, y^{(j)}\rangle = \langle s^{(i)}, y^{(j)}\rangle$. Therefore, we can continue the above series of equation as follows:

$$\sum_{i=1}^{m} w_i = \sum_{p=1}^{l} \beta_p \langle s^{(j)}, y^{(j)}\rangle = \sum_{p=1}^{l} \beta_p(\sum_{i=1}^{m} s_i^{(j)} y_i^{(j)}) = (\sum_{i=1}^{m} w_i^{(j)}) \sum_{p=1}^{l} \beta_p = \sum_{i=1}^{m} w_i^{(j)}.$$

□

In the remaining of this section, we show how to find a weight vector that gives us a robust feasible solution. To do that, we use the w-center for another set $\bar{\mathcal{F}}$. For a vector $\gamma := (\gamma_1, \cdots, \gamma_m)$ we define a new set $\bar{\mathcal{F}}(\gamma)$ like $\mathcal{F}$ as follows

$$\bar{\mathcal{F}}(\gamma) := \left\{ x \in \mathbb{R}^n : \langle a^{(i)}, x\rangle \leq b_i^{(0)} - \gamma_i \sum_{l=1}^{N_i} \Delta b_i^l, \forall i \in \{1, 2, .., m\} \right\}. \tag{3.8}$$

We want to consider the w-center of this set. Like (3.4), the weighted center of $w > 0$ with respect to $\bar{\mathcal{F}}(\gamma)$, i.e., $\{(\bar{x}(w), \bar{y}(w), \bar{s}(w)) : w > 0\}$, is the solution set of the following system of equations and strict inequalities:

$$\begin{aligned}
&\langle a_i, \bar{x}\rangle + \bar{s}_i = b_i^{(0)} - \gamma_i \sum_{l=1}^{N_i} \Delta b_i^l, \ \ \bar{s}_i > 0, \quad i \in \{1, \cdots, m\} \\
&A^T \bar{y} = 0, \\
&\bar{s}_i \bar{y}_i = \bar{w}_i \quad i \in \{1, \cdots, m\}.
\end{aligned} \tag{3.9}$$

We rewrite (3.9) as

$$\begin{aligned}
&\langle a_i, \bar{x}\rangle + (\bar{s}_i + \gamma_i \sum_{l=1}^{N_i} \Delta b_i^l) = b_i^{(0)}, \quad i \in \{1, \cdots, m\} \\
&A^T \bar{y} = 0, \\
&(\bar{s}_i + \gamma_i \sum_{l=1}^{N_i} \Delta b_i^l)\bar{y}_i = \bar{w}_i + \bar{y}_i \gamma_i \sum_{l=1}^{N_i} \Delta b_i^l, \quad i \in \{1, \cdots, m\}.
\end{aligned} \tag{3.10}$$

Let Δb be the vector produced by entries $\sum_{l=1}^{N_i} \Delta b_i^l$, and $\Gamma := \text{Diag}(\gamma)$. From (3.10) we conclude that $(\bar{x}, \bar{y}, \bar{s} + \Gamma \Delta b)$ is the $(\bar{w} + \bar{Y}\Gamma\Delta b)$-center with respect to $\mathcal{F}$, where $\bar{Y} = \text{Diag}(\bar{y})$.

Assume that (x, y, s) is the w-center with respect to $\mathcal{F}$ for a vector w that $\sum_{i=1}^m w_i = 1$. If we use Lemma 3.2.1 for (x, y, s) and $(\bar{x}, \bar{y}, \bar{s} + \Delta\Gamma b)$, for every $\beta \in [0, 1]$, we can construct a weight vector w' with weighted center (x', y', s') such that $y' = y$. For the new weight vector, by using (3.7), we have

$$\begin{aligned} w_i' &= \beta w_i + \frac{y_i}{\bar{y}_i}(1-\beta)(\bar{w}_i + \bar{y}_i\gamma_i \sum_{l=1}^{N_i} \Delta b_i^l) \\ &= \beta w_i + \frac{y_i}{\bar{y}_i}(1-\beta)\bar{w}_i + (1-\beta)y_i\gamma_i\|\Delta b_i\|_1 \quad \forall i \in \{1, 2, \cdots, m\}. \end{aligned} \tag{3.11}$$

We also have, from Lemma 3.2.1, that $\sum_{i=1}^m w_i' = \sum_{i=1}^m w_i = 1$. Let us define $\delta_i := \frac{w_i}{y_i\|\Delta b_i\|_1}$, $\bar{\delta}_i := \frac{\bar{w}_i}{\bar{y}_i\|\Delta b_i\|_1}$. Then from (3.11) we have

$$w_i' = (\beta\delta_i + (1-\beta)\bar{\delta}_i + (1-\beta))y_i\|\Delta b_i\|_1.$$

For $\beta = 1$ we have $w_i' = w_i$ which is equivalent to the randomly chosen weight vector. For $\beta = 0$, we have $w_i' = (\bar{\delta}_i + 1)y_i\|\Delta b_i\|_1 \geq y_i\|\Delta b_i\|_1$ for all $i \in \{1, \cdots, m\}$ and, by definition, $w' \in \mathcal{W}$; we get a weight vector in $\mathcal{W}$ as we wanted. By varying β between 0 and 1, we can control the degree of conservatism.

3.3 Properties of w-space

In this section, we study the structure of the w-space, which is really important for the design of the algorithms in Chapter 6. Let s and y be centric by Definition 3.2.1. First, we note that the simplex of the weight vectors can be divided into regions of y-vector constant (W_y) and s-vector constant (W_s). By using Lemma 3.2.1, if $(\hat{x}, \hat{y}, \hat{s})$ is the solution of system (3.4) corresponding to the weight vector $\hat{w} \in W$, and $\bar{y} > 0$ is any centric y-vector, then $(\hat{x}, \bar{y}, \hat{s})$ is the solution of system (3.4) corresponding to the weight vector $\left(\frac{\bar{y}}{\hat{y}}\hat{w}\right)$. This means that for every centric vector $\hat{s}$ and any centric vector y, $\hat{S}y$ is a weight vector in the simplex.

For every pair of centric vectors s and y, W_s and W_y are convex. To see that, let $(x, \bar{y}, s)$ and (x, y, s) be the weighted centers of $\hat{w}$ and w. Then it is easy to see that for every $\beta \in [0, 1]$, $(x, \beta\bar{y} + (1-\beta)y, s)$ is the weighted center of $\beta\hat{w} + (1-\beta)w$. With a similar reasoning, W_y is convex for every centric y.

By using (3.4), we can write W_s and W_y as follows:

$$\begin{aligned} W_y &= \left\{Y(b - Ax) \ : \ Ax < b, \ y^T(b - Ax) = 1\right\} \\ &= \left\{w > 0 \ : \ YAx + w = Yb, \ e^T w = 1\right\} \\ &= Y[(\mathcal{R}(A) + b) \cap \mathbb{R}_{++}^m] \cap B_1(0, 1), \end{aligned} \tag{3.12}$$

$$\begin{aligned} W_s &= \left\{ Sy \ : \ A^T y = 0, \ y > 0 \ , \ s^T y = 1 \right\} \\ &= \left\{ w > 0 \ : \ A^T S^{-1} w = 0, \ e^T w = 1 \right\} \\ &= S[\mathcal{N}(A^T) \cap \mathbb{R}^m_{++}] \cap B_1(0,1), \end{aligned} \tag{3.13}$$

where $B_1(0,1)$ is the unit ball in 1-norm centered at zero vector. Here we want to find another formulation for W_y that might work better in some cases. To do that, we use the following lemma.

Lemma 3.3.1. *Assume that the rows of $B_y \in \mathbb{R}^{(m-n)\times m}$ make a basis for the null space of $A^T Y$. Then there exists $x \in \mathbb{R}^n$ such that $YAx + w = Yb$ if and only if $B_y w = B_y Yb$. I.e., $(Yb - w) \in \mathcal{R}(YA)$ iff $(Yb - w) \in \mathcal{N}(B_y)$.*

Proof. Assume that there exists x such that $YAx + w = Yb$. By multiplying both sides with B_y from the left and using the fact that $B_y YA = 0$ we have the result. For the other side, assume that $B_y w = B_y Yb$. Then $B_y(w - Yb) = 0$ which means $w - Yb$ is in the null space of B_y. Then, using the orthogonal decomposition theorem, we have $\mathcal{N}(B_y) = \mathcal{R}(B_y^T)^\perp = \mathcal{N}(A^T Y)^\perp = \mathcal{R}(YA)$. So there exists x such that $YAx + w = Yb$. □

Assume that $B \in \mathbb{R}^{(m-n)\times m}$ is such that its rows make a basis for the null space of A^T. For every vector y, we have $A^T y = A^T Y(Y^{-1}y)$, so if y is in the null space of A^T, $Y^{-1}y$ is in the null space of $A^T Y$. Hence, if the rows of B make a basis for the null space of A^T, the rows of BY^{-1} make a basis for the null space of $A^T Y$ and we can write $B_y = BY^{-1}$. By using Lemma 3.3.1, there exists x such that $YAx + w = Yb$ if and only if $BY^{-1}w = BY^{-1}Yb = Bb$, and we can write (3.12) as:

$$\begin{aligned} W_y &= \left\{ w > 0 \ : \ YAx + w = Yb, \ e^T w = 1 \right\} \\ &= \left\{ w > 0 \ : \ BY^{-1} w = Bb, \ e^T w = 1 \right\}. \end{aligned} \tag{3.14}$$

Let us denote the affine hull with aff(.). We can prove the following lemma about W_s and W_y.

Lemma 3.3.2. *Assume that s and y are centric, we have*

$$W_s = \text{aff}(W_s) \cap W \quad \text{and} \quad W_y = \text{aff}(W_y) \cap W.$$

Proof. We prove the first one and our proof for the second one is the same. Clearly we have $W_s \subseteq \text{aff}(W_s) \cap W$. To prove the other side, assume by contradiction that there exist $w \in \text{aff}(W_s) \cap W$ such that $w \notin W_s$. Pick an arbitrary $\hat{w} \in \text{relint}(W_s)$ and consider all the points $w(\beta) = \beta\hat{w} + (1-\beta)w$ for $\beta \in [0,1]$. Both w and $\hat{w}$ are in $\text{aff}(W_s)$, so all the points $w(\beta)$ are also in $\text{aff}(W_s)$. $w(0) \in W_s$ and $w(1) \notin W_s$, so let $\hat{\beta}$ be $\sup\{\beta : w(\beta) \in W_s\}$.

Note that all the points in W_s has the same s-vector, so we have $w(\beta) = Sy(\beta)$ for $\beta \in [0, \hat{\beta})$. By using (3.4) we must also have $w(\hat{\beta}) \in W_s$. We want to prove that $\hat{\beta} = 1$.

Assume that $\hat{\beta} < 1$. All the points on the line segment between $w(0)$ and $w(\hat{\beta})$ have the same s-vector and we can write them as $S(\gamma y(0) + (1-\gamma)y(\hat{\beta}))$ for $\gamma \in [0,1]$. But note that $y(\hat{\beta}) > 0$, so there is a small enough $\epsilon > 0$ such that $y_\epsilon = (-\epsilon y(0) + (1+\epsilon)y(\hat{\beta})) > 0$ and hence Sy_ϵ is a weight vector in W_s. However, it is also a vector on the line segment between $w(\hat{\beta})$ and w which is a contradiction to $\hat{\beta} = \sup\{\beta : w(\beta) \in W_s\}$. So $\hat{\beta} = 1$ and $w = w(1) \in W_s$ which is a contradiction. Hence $W_s \supseteq \mathrm{aff}(W_s) \cap W$ and we are done. $\square$

By the above discussions, we conclude that W is sliced in two ways by W_ys and W_ss for centric s and y vectors. For each centric s and each centric y, W_y and W_s intersect at a single point Sy on the simplex. We want to prove that the smallest affine subspace containing W_s and W_y is the simplex W. To do that, we need to prove some results on the intersection of affine subspaces. We start with the following definition:

Definition 3.3.1. *The **recession cone** of a convex set C in a vector space $\mathbb{E}$ is denoted by $\mathrm{rec}(C)$ and defined as:*

$$\mathrm{rec}(C) := \{y \in \mathbb{E} \ : \ (x+y) \in C, \quad \forall x \in C\}.$$

*The **lineality space** of a convex set C is denoted by $\mathrm{lin}(C)$ and defined as:*

$$\mathrm{lin}(C) := (\mathrm{rec}(C)) \cap (-\mathrm{rec}(C)).$$

Let U be an affine subspace of $\mathbb{R}^m$. It is easy to see that if $y \in \mathrm{rec}(U)$, then $-y \in \mathrm{rec}(U)$, which means $(\mathrm{rec}(U)) = (-\mathrm{rec}(U))$. Therefore, by using Definition 3.3.1 we have $\mathrm{lin}(U) = \mathrm{rec}(U)$. Then, by using the definition of the affine space we have:

$$\mathrm{lin}(U) := \{u_1 - u_2 : \forall u_1, u_2 \in U\}. \tag{3.15}$$

In other words, $\mathrm{lin}(U)$ is a linear subspace such that $U = u + \mathrm{lin}(U)$ for all $u \in U$ where $'+'$ is the Minkowski sum. Now, we have the following lemma, see, for instance, [22].

Lemma 3.3.3. *Given a pair of nonempty affine subspaces U and V in $\mathbb{R}^n$, the following facts hold:*
(1) $U \cap V \neq \emptyset$ iff for every $u \in U$ and $v \in V$, we have $(v-u) \in \mathrm{lin}(U) + \mathrm{lin}(V)$.
(2) $U \cap V$ consists of a single point iff for every $u \in U$ and $v \in V$, we have

$$(v-u) \in \mathrm{lin}(U) + \mathrm{lin}(V) \ \text{ and } \ \mathrm{lin}(U) \cap \mathrm{lin}(V) = \{0\}.$$

(3) For every $u \in U$ and $v \in V$, we have

$$\mathrm{lin}(\mathrm{aff}(U \cup V)) = \mathrm{lin}(U) + \mathrm{lin}(V) + \{\alpha(v-u) : \ \alpha \in \mathbb{R}\}.$$

Proof. (1) For every $u \in U$ and $v \in V$ we have $U = u + \mathrm{lin}(U)$ and $V = v + \mathrm{lin}(V)$. Now, we can write

$$\begin{aligned} U \cap V \neq \emptyset \ &\Leftrightarrow \ 0 \in U - V \\ &\Leftrightarrow \ 0 \in (u - v) + \mathrm{lin}(U) + \mathrm{lin}(V) \\ &\Leftrightarrow \ (u - v) \in \mathrm{lin}(U) + \mathrm{lin}(V). \end{aligned} \tag{3.16}$$

(2) First we prove that if $U \cap V \neq \emptyset$, then $\mathrm{lin}(U \cap V) = \mathrm{lin}(U) \cap \mathrm{lin}(V)$.
If $x \in \mathrm{lin}(U) \cap \mathrm{lin}(V)$, by (3.15), for every $w \in U \cap V$ we have $(w+x) \in U$ and $(w+x) \in V$. Hence for every $w \in U \cap V$, $(w + x) \in U \cap V$ which means $x \in \mathrm{lin}(U \cap V)$. For the other direction, assume that $x \in \mathrm{lin}(U \cap V)$ and let $w \in U \cap V$. Then we have $(w + x) \in U \cap V$, i.e., $(w + x) \in U$ and $(w + x) \in V$. So, we have $x \in \mathrm{lin}(U) \cap \mathrm{lin}(V)$.

To prove (2), note that for every $w \in U \cap V$ we have

$$U \cap V = w + \mathrm{lin}(U \cap V) = w + \mathrm{lin}(U) \cap \mathrm{lin}(V).$$

The result is clear from the last equation and (1).

(3) Let $u \in U$ and $v \in V$ be arbitrary points and let us denote
$\mathrm{lin}(U) + \mathrm{lin}(V) + \{\alpha(v - u) : \ \alpha \in \mathbb{R}\}$ by RHS. Clearly, $\mathrm{lin}(\mathrm{aff}(U \cup V))$ contains $\mathrm{lin}(U)$, $\mathrm{lin}(V)$, and $\{\alpha(v - u) : \ \alpha \in \mathbb{R}\}$ for every $u \in U$ and $v \in V$ and because it is a linear subspace, it contains the direct sum of them, i.e., $RHS \subseteq \mathrm{lin}(\mathrm{aff}(U \cup V))$. To prove $\mathrm{lin}(\mathrm{aff}(U \cup V)) \subseteq RHS$, first we prove that $u + RHS$ contains both U and V. We have $U = u + \mathrm{lin}(U)$ and clearly $U \subseteq u + RHS$. For every $\bar{v} \in V$ we can write:

$$\bar{v} = u + 0 + (\bar{v} - v) + (v - u) \in \ u + RHS,$$

since we have $0 \in \mathrm{lin}(U)$ and $(\bar{v} - v) \in \mathrm{lin}(V)$. Therefore, $V \subseteq u + RHS$. Since $\mathrm{aff}(U \cup V)$ is the smallest affine subspace containing U and V, we must have $\mathrm{aff}(U \cup V) \subseteq u + RHS$ and so $\mathrm{lin}(\mathrm{aff}(U \cup V)) \subseteq RHS$. This concludes that $\mathrm{lin}(\mathrm{aff}(U \cup V)) = RHS$. □

Now, we can prove the following lemma about the dimension of the intersection of affine subspaces.

Lemma 3.3.4. *Let U and V be nonempty affine subspaces in $\mathbb{R}^n$. Then we have the following properties:*

(1) if $U \cap V = \emptyset$, then

$$\dim(\mathrm{aff}(U \cup V)) = \dim(U) + \dim(V) + 1 - \dim(\mathrm{lin}(U) \cap \mathrm{lin}(V)),$$

(2) if $U \cap V \neq \emptyset$, then

$$\dim(\mathrm{aff}(U \cup V)) = \dim(U) + \dim(V) - \dim(U \cap V).$$

Proof. The proof is by using Lemma 3.3.4 and the fact that for every pair of linear subspaces $\text{lin}(U)$ and $\text{lin}(V)$, we have

$$\dim(\text{lin}(U)) + \dim(\text{lin}(V)) = \dim(\text{lin}(U) + \text{lin}(V)) + \dim(\text{lin}(U) \cap \text{lin}(V)). \tag{3.17}$$

(1) Choose arbitrary $u \in U$ and $v \in V$. We have $U \cap V = \emptyset$, so by using Lemma 3.3.3-(1) we have $(v - u) \notin \text{lin}(U) + \text{lin}(V)$. Hence Lemma 3.3.3-(3) and (3.17) result in:

$$\begin{aligned}
\dim(\text{aff}(U \cup V)) &= \dim(\text{lin}(\text{aff}(U \cup V))) = \dim(\text{lin}(U) + \text{lin}(V) + \{\alpha(v-u) : \ \alpha \in \mathbb{R}\}) \\
&= \dim(\text{lin}(U) + \text{lin}(V)) + 1 \\
&= \dim(\text{lin}(U)) + \dim(\text{lin}(V)) + 1 - \dim(\text{lin}(U) \cap \text{lin}(V)) \\
&= \dim(U) + \dim(V) + 1 - \dim(\text{lin}(U) \cap \text{lin}(V)).
\end{aligned}$$

(2) Choose arbitrary $u \in U$ and $v \in V$. We have $U \cap V \neq \emptyset$, so by using Lemma 3.3.4-(1) we have $(v - u) \in \text{lin}(U) + \text{lin}(V)$. We showed in the proof of 3.3.4-(2) that in general if $U \cap V \neq \emptyset$ then $\text{lin}(U \cap V) = \text{lin}(U) \cap \text{lin}(V)$. By using a similar equation we can write:

$$\begin{aligned}
\dim(\text{aff}(U \cup V)) &= \dim(\text{lin}(\text{aff}(U \cup V))) = \dim(\text{lin}(U) + \text{lin}(V) + \{\alpha(v-u) : \ \alpha \in \mathbb{R}\}) \\
&= \dim(\text{lin}(U) + \text{lin}(V)) \\
&= \dim(\text{lin}(U)) + \dim(\text{lin}(V)) - \dim(\text{lin}(U) \cap \text{lin}(V)) \\
&= \dim(U) + \dim(V) - \dim(\text{lin}(U \cap V)) \\
&= \dim(U) + \dim(V) - \dim(U \cap V).
\end{aligned}$$

□

Now, we are ready to prove the following proposition:

Proposition 3.3.1. *Assume that s and y are centric s-vector and y-vector, respectively. Then the smallest affine subspace containing W_s and W_y is the simplex W.*

Proof. We assumed that $A \in \mathbb{R}^{m \times n}$ has full column rank, i.e., $\text{rank}(A) = n \leq m$ and the interior of $\{x : Ax \leq b\}$ is not empty. Let B_s denote the set of all centric s-vectors, i.e., the set of s-vectors for which there exist (x, y, s) satisfies all the equations in (3.4). We claim that $B_s = \{s > 0 : s = b - Ax\}$. For every $s \in \{s > 0 : s = b - Ax\}$, pick an arbitrary $y > 0$ such that $A^T y = 0$. For every scalar α we have $A^T(\alpha y) = 0$, so we can choose α such that $\alpha y^T s = 1$. Hence $(x, \alpha y, s)$ satisfies (3.4) and we conclude that $B_s = \{s > 0 : s = b - Ax\}$. The range of A has dimension n and since B_s is not empty; it is easy to see that the dimension of B_s is also n. Moreover, we have $W_y = YB_s$ and since Y is non-singular, we have $\dim(W_y) = n$.

Now denote by B_y the set of centric y-vectors. By (3.4), we have $A^T y = 0$. The dimension of the null space of A^T is $(n-m)$. In addition, we have to consider the restriction $e^T w = 1$; we have

$$1 = e^T w = e^T(Ys) = s^T y = (b - Ax)^T y = b^T y - x^T A^T y = b^T y.$$

So, we have $b^T y = 1$ for centric y-vectors which reduces the dimension by one (since $b \notin \mathcal{R}(A)$), and $\dim(B_y) = m - n - 1$. We have $W_s = SB_y$ and so by the same explanation $\dim(W_s) = m - n - 1$.

We proved that W_s and W_y intersect at only a single point $w = Sy$, so $\dim(W_s \cap W_y) = 0$. By using Lemma 3.3.4-(2) the dimension of the smallest affine subspace containing W_s and W_y is

$$\dim(W_s) + \dim(W_y) - \dim(W_s \cap W_y) = n + m - n - 1 = m - 1.$$

The dimension of the simplex W is also $m - 1$, so by Lemma 3.3.2 W is the least affine subspace containing W_s and W_y. $\square$

The following simple example makes it clearer that how these sets W_s and W_y look.

Example 3.3.1. *Here, we bring two examples for $m = 3,\ n = 1$. For the first example, let $A := [1\ \ -1\ \ -1]^T$ and $b := [1\ 0\ 0]^T$. By using (3.4), the set of centric s-vectors is $B_s = \{[(1-x),\ x,\ x]^T : x \in (0,1)\}$. The set of centric y-vectors is specified by solving $A^T y = 0$ and $b^T y = 1$, while $y > 0$. We can see that in this example, as shown in Figure 3.2, W_ss are parallel line segments while W_ys are line segments which all intersect at $[1\ 0\ 0]^T$. For the second example, let $A := [1\ \ -1\ 0]^T$ and $b := [1\ 0\ 1]^T$. The set of W_ss and W_ys are shown in Figure 3.3 derived by solving (3.4). As can be seen, this time W_ys are parallel line segments and W_ss are line segments which intersect at the point $[0\ 0\ 1]^T$.*

These examples show that the affine hulls of W_{y^1} and W_{y^2} might not intersect for two centric y-vectors y^1 and y^2. This is also true for the affine hulls of W_{s^1} and W_{s^2} for two centric s-vectors s^1 and s^2.

Example 3.3.2. *For the second example, let $A := [3\ \ -3\ \ -2]^T$ and $b := [1\ 1\ 0]^T$. The set of W_ss and W_ys are shown in Figure 3.4, derived by solving (3.4). In this example, none of W_ys, W_ss, or their affine hulls intersect in a single point.*

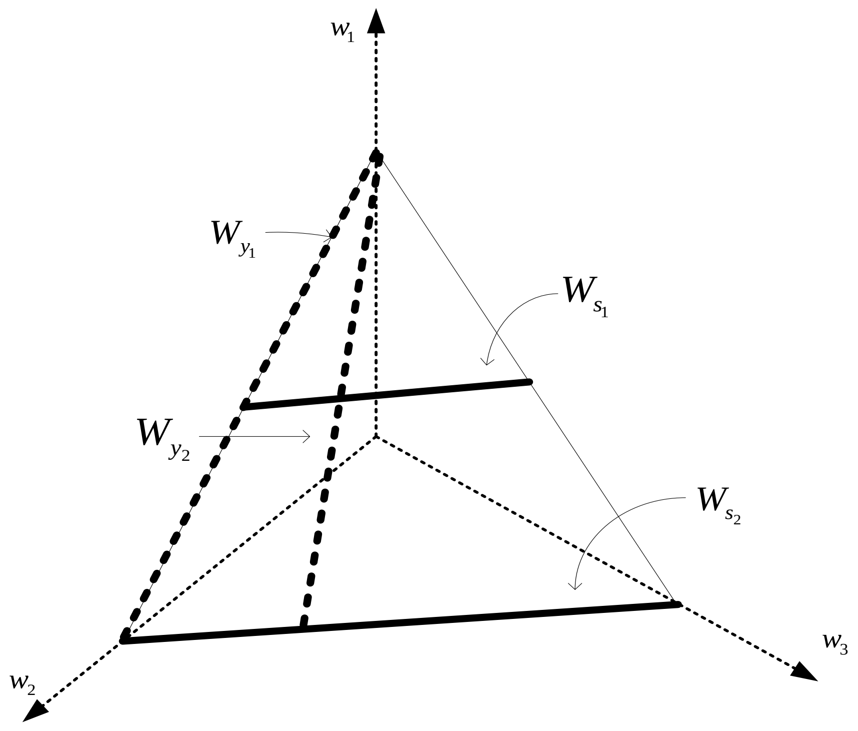

Figure 3.2: W_ss and W_ys for the first example in Example 3.3.1.

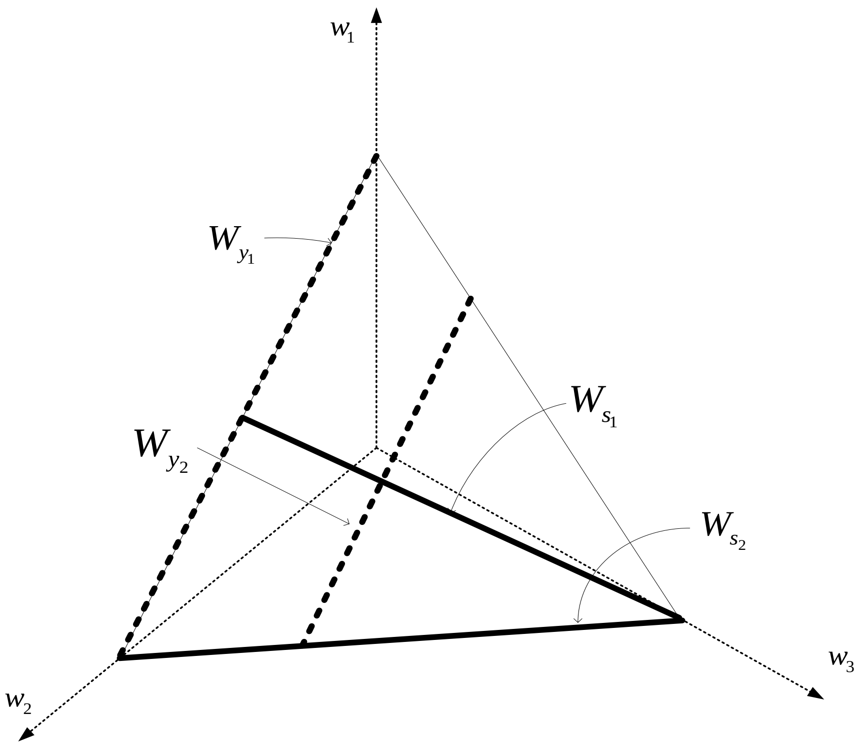

Figure 3.3: W_ss and W_ys for the second example in Example 3.3.1.

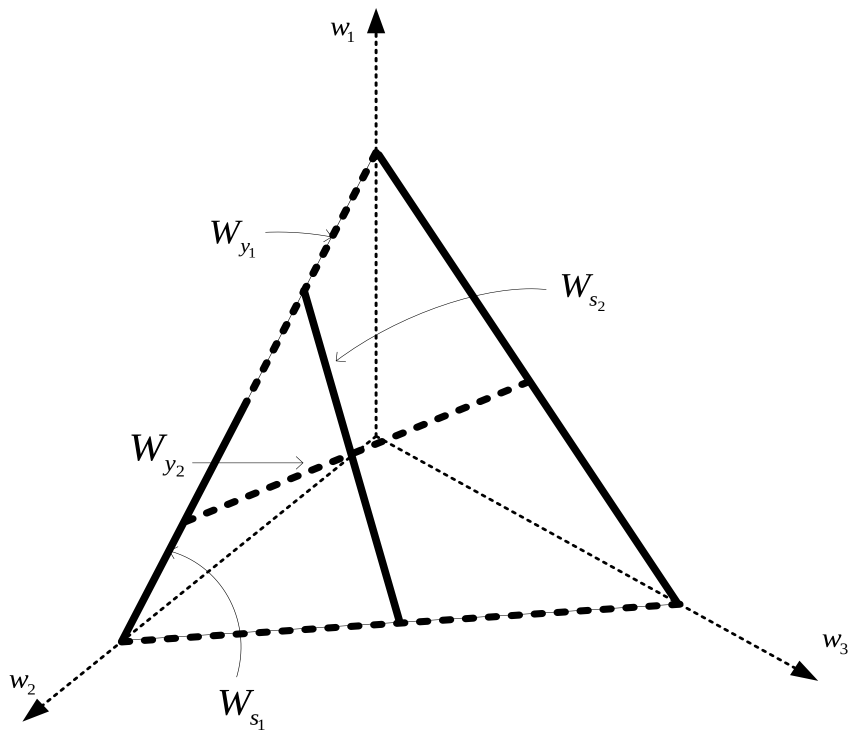

Figure 3.4: W_ss and W_ys for Example 3.3.2.

Chapter 4

Robust Optimization via utility functions

In previous chapters, we introduced our new methodology to deal with LP problems with uncertainty. We explained in Chapter 2 that our approach has many good features in terms of interaction with the decision maker and usability, and its practical advantages over the classical robust optimization are clear. In this chapter, we want to show that many classical robust optimization problems can theoretically be modeled by our approach.

In most of the papers in the robust optimization literature, the uncertainty is considered in the matrix A while we consider it in the RHS. We want to show that by choosing an appropriate utility function $U(s)$ we can model many of the classical robust formulations. In other words, we can find a solution of a classical robust optimization problem by solving

$$\begin{array}{ll} \max & g(x) := U(b - Ax) \\ \text{s.t.} & a_i^T x \leq b_i, \;\; i \in \{1, \cdots, m\}. \end{array} \tag{4.1}$$

Many classical robust optimization models and their approximations can be written as follows

$$\begin{array}{ll} \max & c^T x \\ \text{s.t.} & a_i^T x + f_i(x) \leq b_i, \;\; i \in \{1, \cdots, m\}, \end{array} \tag{4.2}$$

where $f_i(x)$, $i \in \{1, \cdots, m\}$, is a convex function such that $f_i(x) \geq 0$ for all feasible x. By changing $f_i(x)$, different formulations can be derived. In the following we bring some examples. Assume that for each entry A_{ij} of matrix A we have $A_{ij} \in [a_{ij} - \hat{a}_{ij}, a_{ij} + \hat{a}_{ij}]$. It can easily be seen [16] that the classical robust optimization problem is equivalent to (4.2) for $f_i(x) = \hat{a}_i^T |x|$.

For the second example, assume that $A \in \{A : \|M(\text{vec}(A) - \text{vec}(\bar{A}))\| \leq \Delta\}$ for a given $\bar{A}$ where $\|.\|$ is a general norm and M is an invertible matrix. vec(A) is a vector in $\mathbb{R}^{mn\times 1}$ created by stacking the columns of A on top of one another. It is proved in [13] that many approximate robust optimization models can be formulated like this by changing the norm. It is also proved in [13] that this robust optimization model can be formulated as (4.2) by $f_i(x) = \Delta \|M^{-T}x_i\|_*$, where $\|.\|_*$ is the dual norm and $x_i \in \mathbb{R}^{mn\times 1}$ is a vector that contains x in entries $(i-1)n+1$ through in, and 0 everywhere else. Now, utilizing Karush-Kuhn-Tucker (KKT) theorem, we prove that for every robust optimization problem that can be put into form (4.2), there exists a concave utility function U for which (4.1) has the same optimal solution as (4.2).

Theorem 4.0.1. *Assume that* (4.2) *has Slater points. Then, there exists an appropriate concave function $g(x)$ (or equivalently $U(s)$) such that optimization problems* (4.1) *and* (4.2) *have the same optimal solutions.*

Proof. For the optimality condition of (4.2) we have: There exists $\lambda \in \mathbb{R}^m_+$ such that

$$\begin{aligned} &c - \sum_{i=1}^{m} \lambda_i(a_i + \nabla f_i(x)) = 0 \\ &\lambda_i(a_i^T x + f_i(x) - b_i) = 0, \quad i \in \{1, \cdots, m\}. \end{aligned} \tag{4.3}$$

Since the Slater condition holds for (4.2), optimality conditions (4.3) are necessary and sufficient. Let x^* be an optimal solution of (4.2), and let $J \subseteq \{1, \cdots, m\}$ denote the set of indices for which $\lambda_i \neq 0, i \in J$. Let $h(x)$ be an increasing concave function such that its domain contains the positive orthant. We define $g(x)$ as follows:

$$g(x) := c^T x + \sum_{i\in J} \mu_i h(b_i + t_i - a_i^T x - f_i(x)), \tag{4.4}$$

where $t_i > 0, i \in J$ are arbitrary numbers. We claim that $g(x)$ is concave. $b_i + t_i - a_i^T x - f_i(x)$ is a concave function and $h(x)$ is increasing concave, hence $h(b_i + t_i - a_i^T x - f_i(x))$ is a concave function for $i \in \{1, \cdots, m\}$. $g(x)$ is the summation of an affine function and some concave functions and so is concave. The gradient of $g(x)$ is

$$\Rightarrow \nabla g(x) = c - \sum_{i\in J} \mu_i h'(b_i + t_i - a_i^T x - f_i(x))(a_i + \nabla f_i(x)). \tag{4.5}$$

Now choose that $\mu_i, i \in J$ such that

$$\mu_i h'(b_i + t_i - a_i^T x^* - f_i(x^*)) = \lambda_i. \tag{4.6}$$

By comparison of (4.7) and (4.3), we conclude that x^* is a maximizer of $g(x)$ and so is a solution of (4.1), as we wanted. □

For an example, let $h(x) := \ln(x)$, then we have

$$
\begin{aligned}
& g(x) := c^T x + \sum_{i \in J} \mu_i \ln(b_i + t_i - a_i^T x - f_i(x)) \\
\Rightarrow \quad & \nabla g(x) = c - \sum_{i \in J} \frac{\mu_i}{b_i + t_i - a_i^T x - f_i(x)} (a_i + \nabla f_i(x)).
\end{aligned} \tag{4.7}
$$

Therefore, choosing

$$
\frac{\mu_i}{b_i + t_i - a_i^T x^* - f_i(x^*)} = \lambda_i
$$

works. The above explanation proves the existence of a proper utility function. The question is that how can we find such a utility function without having a solution of (4.3). In the following, we will find a function with objective value arbitrarily close to the objective value of (4.2). Assume that strong duality holds for (4.2). Let us define $g(x) := c^T x + \mu \sum_{i=1}^m \ln(b_i - a_i^T x - f_i(x))$ and assume that $\hat{x}$ is the maximizer of $g(x)$. We have

$$
\nabla g(\hat{x}) = c - \sum_{i=1}^m \frac{\mu}{b_i - a_i^T \hat{x} - f_i(\hat{x})} (a_i + \nabla f_i(\hat{x})) = 0. \tag{4.8}
$$

This means that $\hat{x}$ is the maximizer of the Lagrangian of the problem in (4.2), $L(\lambda, x)$, for $\hat{\lambda}_i := \mu / (b_i - a_i^T \hat{x} - f_i(\hat{x}))$, $i \in \{1, \cdots, m\}$. So by strong duality, we have

$$
\begin{aligned}
c^T x^* \leq L(\hat{\lambda}, \hat{x}) \quad &= \quad c^T \hat{x} + \sum_{i=1}^m \frac{\mu}{b_i - a_i^T \hat{x} - f_i(\hat{x})} (b_i - a_i^T \hat{x} - f_i(\hat{x})) \\
&= \quad c^T \hat{x} + m\mu.
\end{aligned} \tag{4.9}
$$

(4.9) shows that by choosing μ small enough, we can construct $g(x)$ such that the optimal objective value of (4.1) is arbitrarily close to the optimal objective value of (4.2).

Chapter 5

Probabilistic Analysis

Probabilistic analysis is tied to robust optimization. One of the recent trends in robust optimization research is the attempt to try reducing conservatism to get better results, and at the same time keeping a good level of robustness. In other words, we have to show that our proposed answer have a low probability of infeasibility. In this chapter, we derive some probability bounds for our algorithms based on weight and slack vectors. To do that, we use the properties derived in Chapter 3. These bounds can be given to the DM with each answer and the DM can use them to improve the next feedback.

5.1 Probability bounds

Assume that we are going to find a robust feasible solution with respect to the uncertainty set $B_1 \times \cdots \times B_m$, where B_i was defined in (3.5). By Proposition 3.2.2, it is equivalent to finding the weighted center for a $w \in \mathcal{W}$, where $\mathcal{W}$ is defined in (3.6). However, finding such a weight vector is not straight forward as we do not have an explicit formula for $\mathcal{W}$. Assume that we pick an arbitrary weight vector $w > 0$ such that $\sum_{i=1}^{m} w_i = 1$, with the weighted center (x, y, s). Let us define the vector δ for w as

$$\delta_i = \frac{w_i}{y_i \|\Delta b_i\|_1}, \quad i \in \{1, 2, \cdots, m\},$$

where Δb_i was defined in (3.5). For each $i \in \{1, \cdots, m\}$, if $1 \le \delta_i$, by Proposition 3.2.2-(b) we have $\langle a_i, x(w) \rangle \le \tilde{b}_i$ for all $\tilde{b}_i \in B_i$. So, the problem is with the constraints that $1 > \delta_i$. For every such constraint, we can find a bound on the probability that $\langle a_j, x(w) \rangle > \tilde{b}_j$. As in the proof of Proposition 3.2.2-(b), in general we can write:

$$\begin{aligned}
\Pr\{\langle a_j, x\rangle > \tilde{b}_j\} &= \Pr\left\{-y_i \sum_{l=1}^{N_i} \Delta b_i^l \ \tilde{z}_i^l > w_i = y_i \delta_i \|\Delta b_i\|_1\right\} \\
&= \Pr\left\{-\sum_{l=1}^{N_i} \Delta b_i^l \ \tilde{z}_i^l > \delta_i \|\Delta b_i\|_1\right\} \\
&\le \exp\left(-\frac{\delta_i^2 (\|\Delta b_i\|_1)^2}{2\sum_{l=1}^{N_i} (\Delta b_i^l)^2}\right). \qquad (5.1)
\end{aligned}$$

where the last inequality is derived by using Hoeffding's inequality which is given in the following lemma:

Lemma 5.1.1. (Hoeffding's inequality[37]) *Let $v_1, v_2, \cdots, v_n$ be independent random variables, and for every $i \in \{1, 2, \cdots, n\}$, $\tau_i \le v_i \le \rho_i$. Then for every $\varphi > 0$*

$$\Pr\left\{\sum_{i=1}^{n} v_i - E\left(\sum_{i=1}^{n} v_i\right\} \ge n\varphi\right] \le \exp\left[\frac{-2n^2\varphi^2}{\sum_{i=1}^{n}(\rho_i - \tau_i)^2}\right].$$

Bertsimas and Sim [16] derived the best possible bound, i.e., a bound that is achievable. The corresponding lemma proved in [16] is as follows:

Lemma 5.1.2. *(a) If $\tilde{z}_i^l$, $l \in \{1, \cdots, N_i\}$, are independent and symmetrically distributed random variables in $[-1, 1]$, p is a positive constant, and $\gamma_{il} \le 1$, $l \in \{1, \cdots, N_i\}$, then*

$$Pr\left\{\sum_{l=1}^{N_i} \gamma_{il} \ \tilde{z}_i^l \ge p\right\} \le B(N_i, p), \qquad (5.2)$$

where

$$B(N_i, p) = \frac{1}{2^{N_i}}\left[(1-\mu)\binom{N_i}{\lfloor \nu \rfloor} + \sum_{i=\lfloor \nu \rfloor + 1}^{N_i} \binom{N_i}{i}\right],$$

where $\nu := (N_i + p)/2$, and $\mu := \nu - \lfloor \nu \rfloor$.
(b) The bound in (5.2) is tight for $\tilde{z}_i^l$ having a discrete probability distribution:
$Pr\{\tilde{z}_i^l = 1\} = Pr\{\tilde{z}_i^l = -1\} = 1/2$, $\gamma_{il} = 1$, $l \in \{1, \cdots, N_i\}$, an integral value of $p \ge 1$, and $p + N_i$ being even.

We can use the bound for our relation (5.1) as follows. Assume that $\tilde{z}_i^l$, $l \in \{1, \cdots, N_i\}$, are independent and symmetrically distributed random variables in $[-1, 1]$. Also denote by $\max(\Delta b_i)$, the maximum entry of Δb_i. Using (5.1), We can write

$$\begin{aligned}
\Pr\{\langle a_j, x\rangle > \tilde{b}_j\} &= \Pr\left\{\sum_{l=1}^{N_i} \Delta b_i^l \; \tilde{z}_i^l > \delta_i \|\Delta b_i\|_1\right\} \\
&\leq \Pr\left\{\sum_{l=1}^{N_i} \frac{\Delta b_i^l}{\max(\Delta b_i)} \; \tilde{z}_i^l \geq \delta_i \frac{\|\Delta b_i\|_1}{\max(\Delta b_i)}\right\} \\
&\leq B\left(N_i, \delta_i \frac{\|\Delta b_i\|_1}{\max(\Delta b_i)}\right).
\end{aligned} \tag{5.3}$$

To compare these two bounds, assume that all the entries of Δb_i are equal. Bound (5.1) reduces to $\exp(-\delta_i^2 N_i/2)$, and bound (5.3) reduces to $B(N_i, \delta_i N_i)$. Figure 5.1 is the comparison of these two bounds for $\delta_i = 0.8$. As can be seen, bound (5.3) dominates bound (5.1). Bound (5.3) is somehow the best possible bound as can be achieved by a special probability distribution as in Lemma 5.1.2.

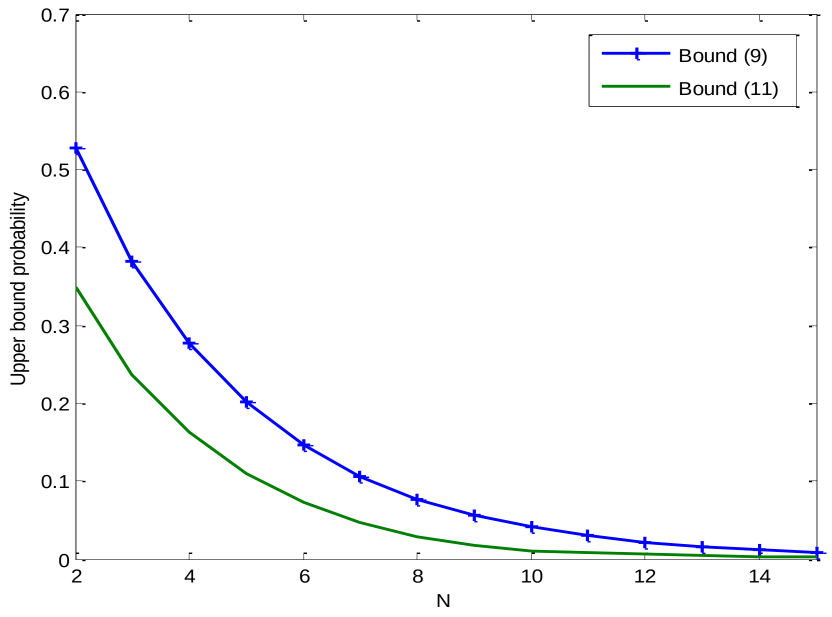

Figure 5.1: Comparison of bounds (5.1) and (5.3).

Now, we want to consider special group of bounds that use the second-order moment using the results in [14]. A similar bound is used by Bertsimas, et al., [13] to find the probability of violation of a single constraint. The main theorem in [14] is

Theorem 5.1.1. *Assume that t is a vector of random variables with mean $\bar{t}$ and covariance matrix Γ. Then for a convex set S we have the following tight bound*

$$\sup_{t\sim(\bar{t},\Gamma)} \Pr\{t \in S\} = \frac{1}{1+d^2},$$

where $d^2 = \inf_{s\in S} \left\{(s-\bar{t})^T\Gamma^{-1}(s-\bar{t})\right\}$.

$\sup_{t\sim(\bar{t},\Gamma)} \Pr\{t \in S\}$ means the tightest upper bound on the probability $\Pr\{t \in S\}$ for all random variables t with mean $\bar{t}$ and covariance matrix Γ. By using Theorem 5.1.1, we can derive the following lemma.

Lemma 5.1.3. *Assume that t is a vector of random variables with mean $\bar{t}$ and covariance matrix Γ. Then for every vector q and any $\tau \geq q^T\bar{t}$ we have*

$$\sup_{t\sim(\bar{t},\Gamma)} \Pr\{q^Tt \geq \tau\} = \frac{q^T\Gamma q}{q^T\Gamma q + (\tau - q^T\bar{t})^2}. \tag{5.4}$$

We can use the bound in Lemma 5.1.3 for our case by substituting $q = \Delta b_i$ and $t = \tilde{z}_i$ in (5.4). Assume that $\tilde{z}_i$ has zero mean and covariance matrix Γ. Then, we can wirte

$$\begin{aligned} \Pr\{\langle a_j, x\rangle > \tilde{b}_j\} &\leq \Pr\left\{\sum_{l=1}^{N_i} \Delta b_i^l \ \tilde{z}_i^l \geq \delta_i \|\Delta b_i\|_1\right\} \\ &\leq \frac{\Delta b_i^T \Gamma \Delta b_i}{\Delta b_i^T \Gamma \Delta b_i + (\delta_i\|\Delta b_i\|_1)^2}. \end{aligned} \tag{5.5}$$

To compare this bound with the two previous ones, assume that all the entries of Δb_i are equal and $\Gamma = \sigma^2 I$. Then bound (5.5) reduces to $\sigma^2/(\sigma^2 + N_i\delta_i^2)$. For small values of σ^2, this bound dominates previous ones, but it is not a good bound for large values of N_i. However, it has the benefit of containing the correlation of the variables in a single constraint.

We can extend this bound to find a bound on the probability of the violation of more than one constraint. To do that, we use the following lemma from [14].

Lemma 5.1.4. *Assume that t is a vector of length k of random variables with mean vector m and covariance matrix Γ. Also assume that ϵ is an arbitrary vector. Then, we have*

$$\sup_{t\sim(\bar{t},\Gamma)} \Pr\{t_i > (1+\epsilon_i)\bar{t}_i \ , \ i \in \{1, \cdots, k\}\} = \frac{1}{1+d^2},$$

where d^2 is given by

$$\begin{aligned} d^2 = \quad &\min \quad s^T\Gamma^{-1}s \\ &\ s.t. \quad\ \ s \geq \bar{T}\epsilon, \end{aligned}$$

where $\bar{T} = \mathrm{Diag}(\bar{t})$. *If* $\Gamma^{-1}\bar{T}\epsilon \geq 0$, *then the tight bound is expressible in closed form:*

$$\sup_{t\sim(\bar{t},\Gamma)} \Pr\{t_i > (1+\epsilon_i)\bar{t}_i \ , \ i \in \{1,\cdots,k\}\} = \frac{1}{1+(\bar{T}\epsilon)^T\Gamma^{-1}\bar{T}\epsilon}.$$

Now, we can consider each t_j for one of our constraints, i.e., $t_j = \sum_{l=1}^{N_i} \Delta b_i^l \ \tilde{z}_i^l$, for $i \in \{1,\cdots,m\}$. If we know the correlation between random variables $\{\tilde{z}_i^l\}$, we can find the correlation between t_i's and thus matrix Γ in Lemma 5.1.4.

As we will see in the next chapter, the above probability bounds do not take part in our algorithm explicitly. However, for each solution, we can present these bounds to the DM and s/he can use them to improve the feedback to the algorithm. As an example on how these bounds can be used for the DM, in the following we show how to construct a concave utility function $U(s)$ based on them.

Bounds (5.1) and (5.3) are functions of $\delta_i = \frac{w_i}{y_i\|\Delta b_i\|_1} = \frac{s_i}{\|\Delta b_i\|_1}$ and as a result, functions of s. Now, assume that based on the probability bounds, the DM defines a function $u_i(s_i)$ for each slack variable s_i as shown in Fig 5.2. $u_i(s_i)$ increases as s_i increases, and then at the point ϵ_i^1 becomes flat. At $s_i = \epsilon_i^2$ it starts to decrease to reach zero. Parameters ϵ_i^1 and ϵ_i^2 are specified by the DM's desired bounds. Now, we can define the utility function as $U(s) := \prod_{j=1}^m u_i(s_i)$. This function is not concave, but maximization of it is equivalent to the maximization of $\ln(U(s))$ which is concave.

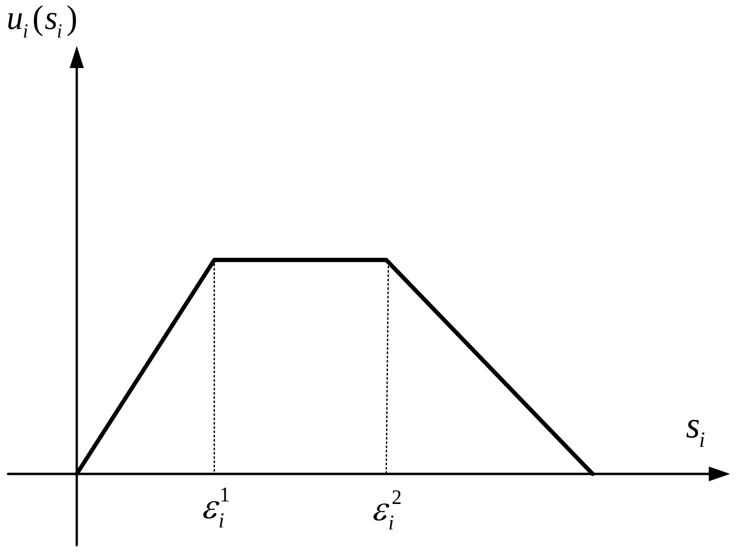

Figure 5.2: The function $u_i(s_i)$ defined for the slack variable s_i

Chapter 6

Algorithms

In Chapter 2 we described our approach and in Chapter 3 we proved some useful properties for the weight space. In this chapter, we develop the cutting-plane algorithms that find an optimal solution for the DM, using the facts we proved in previous chapters.

In the following algorithms, we add a constraint that represents the objective function. This constraint is $\langle c^{(0)}, x\rangle \geq v$, where v is lower bound specified by the information from the DM. For example, if the DM decides that the objective value must not be below a certain value, we can put v equal to that value. So, we change the definition of $\mathcal{F}$ as follows

$$\mathcal{F} := \left\{x \in \mathbb{R}^n : \langle c^{(0)}, x\rangle \geq v,\ \langle a^{(i)}, x\rangle \leq b_i^{(0)}, \forall i \in \{1, 2, .., m\}\right\}. \tag{6.1}$$

We denote the weight vector for the new $\mathcal{F}$ as $w := [w_0, w_1, \cdots, w_m]^T$ where w_0 is related to the $(m+1)^{th}$ constraint represents the objective value. For the start point, as we explained at Section 3.2, we can find a weight vector w' such that $x(w')$ is robust feasible with respect to the worst-case condition. The formula for this weight vector is given in (3.11).

As we mentioned in Chapter 2, assume that the DM has some information about a utility function as a function of the slack variables s, i.e., $U(s)$, and our problem is to maximize this utility function over the set of centric (Definition 3.2.1) s-vectors B_s. So our problem is

$$\begin{aligned} \max \quad & U(s) \\ s.t. \quad & s \in B_s. \end{aligned} \tag{6.2}$$

In the following, we denote an optimal solution of (6.2) with s^{opt}. In many applications, it is possible to capture choices with concave, quasi-concave, or nondecreasing utility functions. We are going to start with the assumption of concave $U(s)$. We will see in Section

8.2 that the algorithm can easily be refined to be used for quasi-concave functions. Here, we can use the concept of supergradient we introduced in Theorem 1.3.2 that satisfies (1.6) for a point x^0. Supergradients (subgradients for convex functions) were used before to design cutting-plane and ellipsoid algorithms. Our goal is to use the concept to design cutting-plane algorithms.

Assume that we start the algorithm from a point $w^0 \in \mathbb{R}^m$ with the corresponding s-vector $s^0 \in \mathbb{R}^m$. By using the idea of supergradient, we can introduce cuts in the s-space or w-space to shrink the set of s-vectors or w-vectors, such that the shrunken space contains an optimal point. In the following sections, we discuss these algorithms in s-space and w-space. Our main algorithm is the one in the w-space, however, the s-space algorithm helps us understand the second one better.

6.1 Cutting-plane algorithm in the s-space

Assume that we have a starting point s^0 and we can obtain a supergradient of U at s^0 from the DM, e.g. g^0, ($g^0 = \nabla U(s^0)$ if U is differentiable at s^0). By using (1.6), for all s such that $U(s) - U(s^0) \geq 0$, we have $(g^0)^T(s - s^0) \geq 0$, i.e.,

$$U(s) - U(s^0) \geq 0 \quad \Rightarrow \quad (g^0)^T(s - s^0) \geq 0. \tag{6.3}$$

This means that all optimal points are in the half-space $(g^0)^T(s - s^0) \geq 0$. So, by adding this cut, we can shrink the s-space and guarantee that there exists an optimal solution in the shrunken part. We can translate this cut to a cut in the x-space by using (3.4):

$$(g^0)^T(s - s^0) = (g^0)^T(b - Ax - b + Ax^0) = (g^0)^T A(x^0 - x).$$

Using this equation, we can consider the cut as a new constraint of the original problem; $(g^0)^T Ax \leq (g^0)^T Ax^0$. Let us define $a^{(m+1)} = (g^0)^T A$ and $b^{(0)}_{m+1} = (g^0)^T Ax^0$. We can redefine $\mathcal{F}$ in (6.1) by adding this new constraint and find the weighted center for a chosen weight vector w^1. The step-by-step algorithm is as follows:

S-space Algorithm:

- Step 1: Set $w = \frac{1}{m}e$ and find the w-centers (x, y, s) with respect to $\mathcal{F}$ and $(\bar{x}, \bar{y}, \bar{s})$ with respect to $\bar{\mathcal{F}}(e)$ defined in (3.8). Set $w^0_i = \frac{y_i}{\bar{y}_i} w_i + y_i \|\Delta b_i\|_1$ for $\forall i \in \{1, 2, \cdots, m\}$ and let (x^0, y^0, s^0) be the w^0-center. x^0 is a robust feasible solution.
- Step 2: Set $k = 0$, $A_0 = A$, $b_0 = b^{(0)}$, and $\mathcal{F}_0 = \mathcal{F}$.
- Step 3: If s^k satisfies the DM, return (x^k, y^k, s^k) and **stop**.

- Step 4: Set $k = k+1$. Find g_{k-1}, the supergradient of $U(s)$ at s^{k-1}. Set

$$A_k = \begin{bmatrix} A_{k-1} \\ g_{k-1}^T A_{k-1} \end{bmatrix}, \qquad b_k^{(0)} = \begin{bmatrix} b_{k-1}^{(0)} \\ g_{k-1}^T A_{k-1} x^{k-1} \end{bmatrix},$$

$$\mathcal{F}_k := \left\{ x \in \mathbb{R}^n : \langle c^{(0)}, x \rangle \geq v, \ \langle a_k^{(i)}, x \rangle \leq (b_k^{(0)})_i, \forall i \in \{1, 2, \cdots, m+k\} \right\} \quad (6.4)$$

- Step 5: Set $w_i^k = \frac{1}{m^2}$ for $i \in \{m+1, \cdots, m+k\}$ and $w_i^k = \frac{1}{m} - \frac{k}{m^2}$ for $i \in \{1, \cdots, m\}$. Find the w^k-center (x^k, y^k, s^k) with respect to $\mathcal{F}_k$. Return to Step 3.

The logic behind Step 5 is that we want to give smaller weights to the new constraints than the original ones. The main problem with the algorithm is that the dimension of the weight space is increased by one every time we add a constraint. We will show that this problem is solved by our w-space algorithm in the following sections.

6.2 Cutting-plane algorithm in the w-space

In this section, we consider the cuts in the w-space. To do that, we first try a natural way of extending the algorithm in the s-space to the one in the w-space. We will show that this extension does not work for all utility functions. Then, we develop an algorithm applicable to all concave utility functions.

Like the s-space, we try to use the supergradients of $U(s)$. Let U_w denote the utility function as a function of w. From (3.4) we have $Ys = w$; so, $U_w(w) = U(s) = U(Y^{-1}w)$. If Y was constant for all weight vectors, $U_w(w)$ would be a concave function, and we could use Lemma 1.3.1 to find the supergradient at each point. The problem here is that Y is not necessarily the same for different weight vectors. Assume that we confine ourselves to weight vectors in the simplex W with the same y-vector (W_y). $U_w(w)$ is a concave function on W_y, so, we can define its supergradient. By Lemma 1.3.1, we conclude that $\partial U_w(w) = Y^{-1} \partial U(s)$ for all $w \in W_y$.

Suppose we start at w^0 with the weighted center (x^0, y^0, s^0). Let us define $g^{0w} := (Y^0)^{-1} g^0$, where g_0 is a supergradient of $U(s)$ at s^0. Then from (1.6) we have,

$$U_w(w) \leq U_w(w^0) + (g^{0w})^T (w - w^0), \quad \forall w \in W_{y_0}. \quad (6.5)$$

If we confine the weight space to W_y, by the same procedure used for s-space, we can introduce cuts in the w-space by using (6.5). The problem is that we do not have a proper characterization of W_y. On the other hand, U_w may not be a concave function on the whole simplex. This is an unsolved problem in [34] that we handle in this chapter.

Assume that s^{opt} is an optimal solution of (6.2), and $W_{s^{opt}}$ is the set of weight vectors in the simplex with s-vector s^{opt}. It is easy to see that $W_{s^{opt}}$ is convex. We also have the following lemma:

Lemma 6.2.1. *Let (x', y', s') be the weighted center corresponding to w', s^{opt} be an optimal solution of (6.2) , and g' be the supergradient of $U(s)$ at s'. Then $S^{opt}y'$ is in the half-space ${g'}_w^T(w - w') \geq 0$, where $g'_w = Y'^{-1}g'$.*

Proof. We have

$$
{g'}_w^T(S^{opt}y' - w') = g'^T Y'^{-1}(S^{opt}y' - S'y') = g'^T(s^{opt} - s') \geq 0.
$$

The last inequality follows from the fact that s^{opt} is a maximizer and g' is a supergradient of $U(s)$ at s'. □

The above lemma shows that using hyperplanes of the form $g'^T Y'^{-1}(w - w')$, we can always keep a point from $W_{s^{opt}}$. Now, by using the fact that $W_{s^{opt}}$ is convex and the above lemma, the question is: if we use a sequence of these hyperplanes, can we always keep a point from $W_{s^{opt}}$?

This question is equivalent to the following one: We start with w^0 and shrink the simplex W into the intersection of the half-space $(g^{0w})^T(w - w^0) \geq 0$ and the simplex, say W_0. Then we choose an arbitrary weight vector w^1 with weighted center (x^1, y^1, s^1) from the shrunken space W_0. If g^1 is a supergradient of $U(s)$ at s^1, then we shrink W_0 into the intersection of W_0 and the half-space $(g^{1w})^T(w - w^1) \geq 0$, where $g^{1w} = (Y^1)^{-1}g^1$, and call the last shrunken space W_1. Is it always true that a weight vector with s-vector s^{opt} exists in W_1?

In the following, we will show that this is true for some utility functions, but not true in general. We define a special set of functions that have good properties for cuts in the w-space, and the above algorithm works for them.

Definition 6.2.1. *A function $f : \mathbb{R}^m_{++} \to \mathbb{R}$ is called Non-decreasing under affine scaling (NDAS) if for every $d \in \mathbb{R}^m_{++}$ we have:*

1. *$f(s) \leq \max\{f(Ds), f(D^{-1}s)\}, \quad \forall s \in \mathbb{R}^m_{++}$.*
2. *If for a single $s^0 \in \mathbb{R}^m_{++}$ we have $f(s^0) \leq f(Ds^0)$, then $f(s) \leq f(Ds)$ for all $s \in \mathbb{R}^m_{++}$.*

As an example of an NDAS function, for every $t \in \mathbb{R}^m$ the function $f_1(s) := \sum_{i=1}^m t_i \log s_i$ is NDAS. To see that, for every $s, d \in \mathbb{R}^m_{++}$ we have:

$$
\begin{aligned}
f_1(s) - f_1(Ds) &= -\sum_{i=1}^m t_i \log d_i, \\
f_1(s) - f_1(D^{-1}s) &= -\sum_{i=1}^m t_i \log \frac{1}{d_i} = \sum_{i=1}^m t_i \log d_i,
\end{aligned}
$$

and so we have $2f_1(s) = f_1(Ds) + f_1(D^{-1}s)$. The second property is also easy to verify and the function is NDAS. $f_1(s)$ is specially important because of its relation to a family of classical utility functions in mathematical economics; Cobb-Douglas production function which is defined as $U_{cd}(s) = \prod_{i=1}^m s_i^{t_i}$, where $t \in \mathbb{R}^m_{++}$. Using this function to simulate problems in economics goes back to 1920's. Maximization of $U_{cd}(s)$ is equivalent to the maximization of its logarithm which is equal to $f_1(s) = \ln(U_{cd}(s)) = \sum_{i=1}^m t_i \log s_i$.

Authors in [36] considered Cobb-Douglas utility function to present an algorithm for evaluating and ranking items with multiple attributes. [36] is related to our work as the proposed algorithm is a cutting-plane one. [36] also used the idea of weight space as the utility function is the weighted sum of the attributes. However, our algorithm uses the concept of weighted analytic center which is completely different.

Now, we have the following proposition.

Proposition 6.2.1. *Assume that $U(s)$ is a NDAS concave function. Let (x^0, y^0, s^0) and (x^1, y^1, s^1) be the weighted centers of w^0 and w^1, and g^0 and g^1 be the supergradients of $U(s)$ at s^0 and s^1, respectively. Then we have*

$$\left\{w : \ (g^{0w})^T(w - w^0) \geq 0, \ (g^{1w})^T(w - w^1) \geq 0\right\} \cap W_{s^{opt}} \neq \ \phi,$$

where $g^{0w} = (Y^0)^{-1}g^0$ and $g^{1w} = (Y^1)^{-1}g^1$.

Proof. Consider the weight vectors $Y^0 s^{opt}$ and $Y^1 s^{opt}$. Our two hyperplanes are

$$P_0 := \{w : \ \ (g^0)^T(Y^0)^{-1}(w - Y^0 s^0) = 0\},$$
$$P_1 := \{w : \ \ (g^1)^T(Y^1)^{-1}(w - Y^1 s^1) = 0\}.$$

By Lemma 6.2.1, $Y^0 s^{opt}$ is in the half-space $(g^0)^T(Y^0)^{-1}(w - Y^0 s^0) \geq 0$ and $Y^1 s^{opt}$ is in the half-space $(g^1)^T(Y^1)^{-1}(w - Y^1 s_1) \geq 0$. If one of these two points is also in the other half-space, then we are done. So by contradiction assume that

$$(g^0)^T(Y^0)^{-1}(Y^1 s^{opt} - Y^0 s^0) < 0 \ \text{ and } \ (g^1)^T(Y^1)^{-1}(Y^0 s^{opt} - Y^1 s^1) < 0,$$

which is equivalent to

$$(g^0)^T((Y^0)^{-1}Y^1 s^{opt} - s^0) < 0 \ \text{ and } \ (g^1)^T((Y^1)^{-1}Y^0 s^{opt} - s^1) < 0. \tag{6.6}$$

Using (6.5) and (6.6) we conclude that

$$U((Y^0)^{-1}Y^1 s^{opt}) < U(s^0) \leq U(s^{opt}) \ \text{ and}$$
$$U((Y^1)^{-1}(Y^0)s^{opt}) < U(s^1) \leq U(s^{opt}).$$

However, note that $(Y^0)^{-1}Y^1 = ((Y^1)^{-1}Y^0)^{-1}$ and this is a contradiction to Definition 6.2.1. So (6.6) is not true and at least one of $Y^0 s^{opt}$ and $Y^1 s^{opt}$ is in

$$\{w : \ (g^{0w})^T(w - w^0) \geq 0, \ (g^{1w})^T(w - w^1) \geq 0\}.$$

$\square$

In Proposition 6.2.1, we proved that by the first two hyperplanes, the intersection of the shrunken space and $W_{s^{opt}}$ is not empty. Now, we want to show that we can continue shrinking the space and have nonempty intersection with $W_{s^{opt}}$.

Proposition 6.2.2. *Assume that $U(s)$ is a NDAS concave function. Let (x^i, y^i, s^i) be the weighted centers of w^i, $i \in \{0, \cdots, k\}$, and g^i be the supergradients of $U(s)$ at s^i. Let us define*

$$W^i := \left\{w : \ (g^{iw})^T(w - w^i) \geq 0\right\} \cap W,$$

where $g^{iw} = (Y^i)^{-1}g^i$. Assume we pick the points such that

$$w^i \in \operatorname{relint}\left(\bigcap_{j=0}^{i-1} W^j\right), \qquad i \in \{1, \cdots, k\}. \tag{6.7}$$

Then we have

$$\left(\bigcap_{j=0}^{k} W^j\right) \ \cap \ W_{s^{opt}} \neq \ \phi, \tag{6.8}$$

where s^{opt} is an optimal solution of (6.2).

Proof. Among the hree representations of W_s were given in (3.13), we use the second one in the following. If (6.8) is not true, then the following system is infeasible:

$$\begin{aligned} &A^T(S^{opt})^{-1}w = 0, \quad e^Tw = 1, \quad w \geq 0, \\ &(g^{iw})^T(w - w^i) \geq 0, \quad i \in \{0, \cdots, k\}. \end{aligned} \tag{6.9}$$

By using Farkas' Lemma, there exist $v \in \mathbb{R}^n$, $p \in \mathbb{R}$, and $q \in \mathbb{R}^k_+$ such that:

$$\begin{aligned} &(S^{opt})^{-1}Av + pe - \sum_{i=0}^{k} q_i g^{iw} \geq 0 \quad \equiv \quad Av + ps^{opt} - \sum_{i=0}^{k} q_i S^{opt}(Y^i)^{-1}g^i \geq 0, \\ &p - \sum_{i=0}^{k} q_i (g^{iw})^T w^i < 0 \quad \equiv \quad p - \sum_{i=0}^{k} q_i (g^i)^T s^i < 0. \end{aligned} \tag{6.10}$$

Now for each $j \in \{0, \cdots, k\}$, we multiply both sides of the first inequality in (6.10) with e^TY^j, then we have:

$$\begin{aligned} &p - \sum_{i=0}^{k} q_i (s^{opt})^T Y^j (Y^i)^{-1} g^i \geq 0, \quad \forall j \in \{0, \cdots, k\}, \\ &p - \sum_{i=0}^{k} q_i (g^i)^T s^i < 0, \end{aligned} \tag{6.11}$$

where we used the facts that $e^T Y^j Av = (A^T y^j)^T v = 0$ and $e^T Y^j s^{opt} = 1$. If we multiply the first set of inequalities in (6.11) with -1 and add it to the second one we have

$$q_j (g^j)^T (s^{opt} - s^j) + \sum_{i \neq j} q_i (g^i)^T (Y^j (Y^i)^{-1} s^{opt} - s^i) < 0, \tag{6.12}$$

for all $j \in \{0, \cdots, k\}$. $q \in \mathbb{R}^k_+$ and $(g^j)^T (s^{opt} - s^j) \geq 0$ by supergradient inequality. Hence, from (6.12), for each $j \in \{0, \cdots, k\}$, there exists $\phi_j \in \{0, \cdots, k\} \backslash \{j\}$ such that $(g^{\phi_j})^T (Y^j (Y^{\phi_j})^{-1} s^{opt} - s^{\phi_j}) < 0$ which, using (6.3), means $U(Y^j (Y^{\phi_j})^{-1} s^{opt}) < U(s^{\phi_j}) \leq U(s^{opt})$. Therefore, by the first property of NDAS functions, we must have

$$U(Y^{\phi_j} (Y^j)^{-1} s^{opt}) \geq U(s^{opt}). \tag{6.13}$$

Now, it is easy to see that there exists a sequence $j_1, \cdots, j_t \in \{0, \cdots, k\}$ such that $\phi_{j_i} = j_{i+1}$ and $\phi_{j_t} = j_1$. By using (6.13) and the second property of NDAS functions $t-1$ times we can write:

$$\begin{aligned} U(s^{opt}) &\leq U(Y^{j_2} (Y^{j_1})^{-1} s^{opt}) \\ &\leq U(Y^{j_3} (Y^{j_2})^{-1} Y^{j_2} (Y^{j_1})^{-1} s^{opt}) \\ &\leq \cdots \leq U(Y^{j_t} (Y^{j_{t-1}})^{-1} \cdots Y^{j_2} (Y^{j_1})^{-1} s^{opt}) \\ &= U(Y^{j_t} (Y^{j_1})^{-1} s^{opt}). \end{aligned} \tag{6.14}$$

However, we had $U(Y^{j_t} (Y^{j_1})^{-1} s^{opt}) = U(Y^{j_t} (Y^{\phi_{j_t}})^{-1} s^{opt}) < U(s^{opt})$ which is a contradiction to (6.14). This means the system (6.9) is feasible and we are done. □

Proposition 6.2.2 shows that the abovementioned cutting-plane algorithm works for the NDAS functions.

We can derive stronger results by using (6.6) in the proof of Proposition 6.2.1. Using (6.6), $(g^0)^T (s^{opt} - s^0) \geq 0$, and $(g^1)^T (s^{opt} - s^1) \geq 0$ we have:

$$(g^0)^T ((I - (Y^0)^{-1} Y^1) s^{opt}) > 0 \quad \text{and} \quad (g^1)^T ((I - (Y^1)^{-1} Y^0) s^{opt}) > 0. \tag{6.15}$$

Now from (6.6), both hyperplanes P_0 and P_1 cut the line segment $[Y^0 s^{opt}, Y^1 s^{opt}]$ in the relative interior of the line segment, and neither of them contains this line segment. So, there exists a set of points on the line segment $[Y^0 s^{opt}, Y^1 s^{opt}]$ that for every point w on it we have:

$$(g^0)^T (Y^0)^{-1} (w - Y^0 s^0) < 0 \quad \text{and} \quad (g^1)^T (Y^1)^{-1} (w - Y^1 s^1) < 0.$$

Let $(\beta Y^0 + (1 - \beta Y^1) s^{opt})$ be one of those points. Then we have

$$\begin{aligned} &(g^0)^T ((\beta I + (1-\beta)(Y^0)^{-1} Y^1) s^{opt} - s^0) < 0 \quad \text{and} \\ &(g^1)^T ((\beta (Y^1)^{-1} Y^0 + (1-\beta) I) s^{opt} - s^1) < 0. \end{aligned} \tag{6.16}$$

So we have:

$$\begin{aligned} &U(s^0) > U((\beta I + (1-\beta)(Y^0)^{-1}Y^1)s^{opt}) \geq \beta U(s^{opt}) + (1-\beta)U((Y^0)^{-1}Y^1 s^{opt}), \text{ and} \\ &U(s^1) > U((\beta (Y^1)^{-1}Y^0 + (1-\beta)I)s^{opt}) \geq \beta U((Y^1)^{-1}Y^0 s^{opt}) + (1-\beta)U(s^{opt}). \end{aligned} \tag{6.17}$$

It would be very helpful in designing a cutting-plane algorithm in the w-space if Proposition 6.2.1 were true in general. However, we show in the following example that it is not true.

Example 6.2.1. *The statement of Proposition 6.2.1 is not true for a general concave function.*

Proof. Consider the first example of Example 3.3.1. We have $m = 3$, $n = 1$, $A = [1,\ -1,\ -1]^T$, and $b = [1,\ 0,\ 0]^T$. Using (3.4), the set of centric s-vectors is

$$B_s = \{[1-x,\ \ x,\ \ x]^T : x \in (0,1)\}.$$

The set of centric y-vectors, B_y, is specified by solving $A^T y = 0$ and $y^T b = 1$ while $y > 0$ and we can see that $B_y = \{[1,\ \ z,\ \ 1-z]^T : z \in (0,1)\}$. As shown in Figure 3.2, W_ss are parallel line segments while W_ys are line segments that all intersect at $[1,\ 0,\ 0]^T$.

Now, assume that the function $U(s)$ is as follows (does not depend on s_3)

$$U(s) = \begin{cases} 3s_1 - s_2, & \text{if} \quad s_1 \leq s_2; \\ -s_1 + 3s_2, & \text{if} \quad s_1 > s_2. \end{cases} \tag{6.18}$$

This function is piecewise linear and it is easy to see that it is concave. $U(s)$ is also differentiable at all the points except the points $s_1 = s_2$. At any point that the function is differentiable, the supergradient is equal to the gradient of the function at that point. Hence, we have $\partial U(s) = \{[3,\ -1,\ 0]^T\}$ for $s_1 < s_2$ and $\partial U(s) = \{[-1,\ 3,\ \ 0]^T\}$ for $s_1 > s_2$.

If we consider $U(s)$ on B_s, we can see that the maximum of the function is attained at the point that $s_1 = s_2$, so $s_{opt} = [1/2,\ 1/2,\ 1/2]^T$. Now assume that we start at $w^0 = S^0 y^0 = [0.4,\ 0.1,\ 0.5]^T$. Because we have $y_1 = 1$ for all centric y-vectors, $w_1 = s_1$, and we can easily find s^0 and y^0 as $s^0 = [0.4,\ 0.6,\ 0.6]^T$ and $y^0 = [1,\ 1/6,\ 5/6]^T$. The hyperplane passing through w^0 is $(g^0)^T(Y^0)^{-1}(w - w^0) = 0$ and since $s^0_1 < s^0_2$ we have

$$(g^0)^T(Y^0)^{-1} = [3,\ -1,\ \ 0](Y^0)^{-1} = [3,\ -6,\ \ 0], \tag{6.19}$$

and we can write the hyperplane as $3(w_1 - 0.4) - 6(w_2 - 0.1) = 0$. In the next step, we have to choose a point w^1 such that $(g^0)^T(Y^0)^{-1}(w^1 - w^0) \geq 0$. Let us pick $w^1 = [0.6,\ 0.19,\ 0.21]^T$ for which we can easily find $s^1 = [0.6,\ 0.4,\ 0.4]^T$ and $y^1 = [1,\ 0.475,\ 0.525]^T$. For this point we have $s^1_1 > s^2_2$, so $(g^1)^T(Y^1)^{-1} = [-1,\ 6.32,\ 0]^T$ and the hyperplane passing through w^1

is $-(w_1 - 0.6) + 6.32(w_2 - 0.19) = 0$. The intersection of two hyperplanes on the simplex can be found by solving the following system of equations:

$$\begin{cases} 3w_1 - 6w_2 = 0.6 \\ -w_1 - 6w_2 = 0.6 \\ w_1 + w_2 + w_3 = 1 \end{cases} \Rightarrow \ w^* = \begin{bmatrix} 0.57 \\ 0.185 \\ 0.245 \end{bmatrix}. \qquad (6.20)$$

The intersection of simplex and the hyperplanes $(g^0)^T(Y^0)^{-1}(w - w^0) = 0$ and $(g^1)^T(Y^1)^{-1}(w - w^1) = 0$ are shown in Fig 6.1. The intersection of simplex with $\{w : \ (g^0)^T(Y^0)^{-1}(w - w^0) \geq 0, \ (g^1)^T(Y^1)^{-1}(w - w^1) \geq 0\}$ is shown by hatching lines. As can be seen, we have:

$$\left\{w : \ (g^{0w})^T(w - w^0) \geq 0, \ (g^{1w})^T(w - w^1) \geq 0\right\} \cap W_{s_{op}} = \ \phi.$$

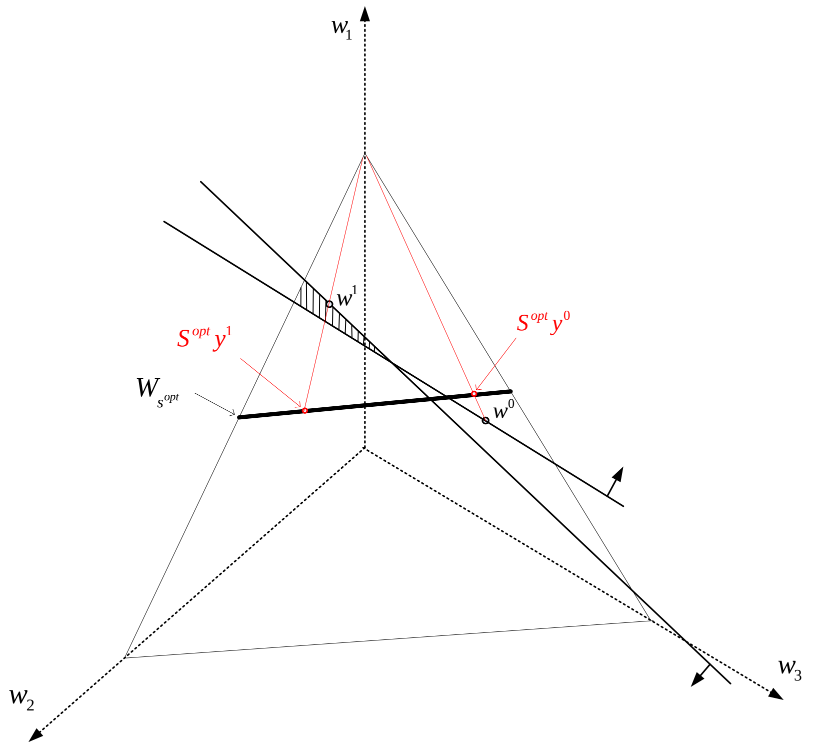

Figure 6.1: Intersection of simplex and the hyperplanes $(g^0)^T(Y^0)^{-1}(w - w^0) = 0$ and $(g^1)^T(Y^1)^{-1}(w - w^1) = 0$ in Example 6.2.1.

$\square$

Example 6.2.1 shows that the statement of Theorem 6.2.1 is not true for a general concave function $U(s)$. To be able to perform a cutting-plane algorithm in the w-space, we have to modify the definition of cutting hyperplanes. In the next two propositions, we introduce a new set of cutting-planes.

Proposition 6.2.3. *For every point $Y^0s^0 \in W$, there exists a hyperplane P passing through it such that:*
1- P contains all the points in W_{s^0}, and
2- P cuts W_{y^0} the same way as $(g^0)^T(Y^0)^{-1}(w - Y^0s^0) = 0$ cuts it; this is, the intersections of P and $(g^0)^T(Y^0)^{-1}(w - Y^0s^0) = 0$ with W_{y^0} is the same, and the projections of their normals onto W_{y^0} have the same direction.

Proof. Assume that $w^0 = Y^0s^0$ is the point that is chosen and let u^0 be the normal vector to the desired hyperplane P. First, we want the hyperplane to contain W_{s^0}. This means that for all centric $\hat{y}$, the vector $S^0y^0 - S^0\hat{y}$ is on P, i.e., we have $(u^0)^TS^0(y^0 - \hat{y}) = 0$. Since $A^T(y^0 - \hat{y}) = 0$, we can put $u^0 = (S^0)^{-1}Ah^0$ with an arbitrary h^0 and we have:

$$(u^0)^TS^0(y^0 - \hat{y}) = (h^0)^TA^T(S^0)^{-1}S^0(y^0 - \hat{y}) = 0.$$

Now, we want to find h^0 such that $(u^0)^T(w - Y^0s^0)$ cuts W_{y^0} the same way as $(g^0)^T(Y^0)^{-1}(w - Y^0s^0)$ cuts it. We actually want to find h^0 which satisfies the stronger property that $(u^0)^T(w - Y^0s^0) = (g^0)^T(Y_0)^{-1}(w - Y^0s^0)$ for all $w \in W_{y^0}$. All the points in W_{y^0} are of the form $Y^0\hat{s}$, so we must have $(u^0)^TY^0(\hat{s} - s^0) = (g^0)^T(\hat{s} - s^0)$. Since $(\hat{s} - s^0)$ is in the range of A, we have:

$$(u^0)^TY^0Ax = (g^0)^TAx \;\Rightarrow\; ((u^0)^TY^0 - (g^0)^T)Ax = 0, \quad \forall x \in \mathbb{R}^n.$$

This means that $Y^0u^0 - g^0$ must be in the $\mathcal{R}(A)^\perp = \mathcal{N}(A^T)$, which means $A^T(Y^0u^0 - g^0) = 0$. However, we had from above that $u^0 = (S^0)^{-1}Ah^0$ and hence:

$$A^TY^0u^0 = A^Tg^0 \;\Rightarrow\; A^TY^0(S^0)^{-1}Ah^0 = A^Tg^0 \;\Rightarrow\; h^0 = (A^TY^0(S^0)^{-1}A)^{-1}A^Tg^0. \quad (6.21)$$

So, the hyperplane with normal vector $u^0 = (S^0)^{-1}Ah^0$, where $h^0 = (A^TY^0(S^0)^{-1}A)^{-1}A^Tg^0$ has the required properties. Because this hyperplane cuts W_{y^0} the same way as $(g^0)^T(Y^0)^{-1}(w - Y^0s^0)$ does, we conclude that $(u^0)^T(Y^0s^{opt} - Y^0s^0) \geq 0$. This means that Y^0s^{opt} is in the half-space $(u^0)^T(w - Y^0s^0) \geq 0$. □

The normal of the hyperplane derived in Proposition 6.2.3 has a nice interpretation with respect to orthogonal projection and the primal-dual scaling $Y^{-1}S$. We have:

$$\begin{aligned} u^0 &= (S^0)^{-1}A(A^TY^0(S^0)^{-1}A)^{-1}A^Tg^0 \\ &= (Y^0)^{-1/2}(S^0)^{-1/2} \\ &\qquad \underbrace{[((Y^0)^{1/2}(S^0)^{-1/2}A)(A^TY^0(S^0)^{-1}A)^{-1}(A^T(S^0)^{-1/2}(Y^0)^{1/2})]}_{P}(Y^0)^{-1/2}(S^0)^{1/2}g_0 \\ &= (Y^0)^{-1/2}(S^0)^{-1/2}P(Y^0)^{-1/2}(S^0)^{1/2}g_0, \qquad (6.22) \end{aligned}$$

where P is the orthogonal projection onto the range of $(Y^0)^{1/2}(S^0)^{-1/2}A$.

Note that a main benefit of the hyperplane in Proposition 6.2.3 is that when we choose a point, we can cut away all the points with the same s-vector. Now, we prove the following proposition which shows we can cut the simplex with a sequence of hyperplanes such that the intersection of their corresponding half-spaces contain a point from $W_{s^{opt}}$.

Proposition 6.2.4. *Assume that we choose the points $Y^0s^0, Y^1s^1 \in W$. There exists a hyperplane P with the normal vector u^1 passing through Y^1s^1 such that:*
1- P contains all the points in W_{s^1}, and
2- $(u^1)^T(Y^0s^{opt} - Y^1s^1) \geq 0$ for every feasible maximizer of $U(s)$.

Proof. As in the proof of Proposition 6.2.3, if we set $u^1 = (S^1)^{-1}Ah^1$, then the hyperplane contains all the points in W_{s^1}. To satisfy the second property, we want to find h^1 with the stronger property that

$$(u^1)^T(Y^0\hat{s} - Y^1s^1) = (g^1)^T(\hat{s} - s^1), \tag{6.23}$$

for all the centric $\hat{s}$. The reason is that we already have $(g^1)^T(s_{opt} - s^1) \geq 0$. By the choice of $u^1 = (S^1)^{-1}Ah^1$, for every centric y we have

$$(u^1)^TS^1y = (h^1)^TA^T(S^1)^{-1}S^1y = (h^1)^TA^Ty = 0.$$

So, we have $(u^1)^TY^1s^1 = (u^1)^TY^0s^1 = 0$ and we can continue the above equation as follows:

$$\begin{aligned}(g^1)^T(\hat{s} - s^1) &= (u^1)^T(Y^0\hat{s} - Y^1s^1) = (u^1)^T(Y^0\hat{s}) \\ &= (u^1)^T(Y^0\hat{s} - Y^0s^1) \\ &= (u^1)^TY^0(\hat{s} - s^1).\end{aligned}$$

Now we can continue in a similar way as in the proof of Proposition 6.2.3. Since $(\hat{s} - s^0)$ is in the range of A, we must have:

$$((u^1)^TY^0 - (g^1)^T)Ax = 0, \quad \forall x \in \mathbb{R}^n.$$

By the same reasoning, we have:

$$A^TY^0u^1 = A^Tg^1 \;\Rightarrow\; A^TY^0(S^1)^{-1}Ah^1 = A^Tg^1 \;\Rightarrow\; h^1 = (A^TY^0(S^1)^{-1}A)^{-1}A^Tg^1. \tag{6.24}$$

So, the hyperplane with normal vector $u^1 = (S^0)^{-1}Ah^1$, where $h^1 = (A^TY^0(S^1)^{-1}A)^{-1}A^Tg^1$ has the required properties. □

By Proposition 6.2.4, we can create a sequence of points and hyperplanes such that the corresponding half-spaces contain Y^0s^{opt}. The algorithm is as follows:

W-space Algorithm:

- Step 1: Set $w = \frac{1}{m}e$ and find the w-centers (x, y, s) with respect to $\mathcal{F}$ and $(\bar{x}, \bar{y}, \bar{s})$ with respect to $\bar{\mathcal{F}}(e)$ defined in (3.8). Set $w_i^0 = \frac{y_i}{\bar{y}_i} w_i + y_i \|\Delta b_i\|_1$ for $\forall i \in \{1, 2, \cdots, m\}$ and let (x^0, y^0, s^0) be the w^0-center. x^0 is a robust feasible solution.

- Step 2: Set $k = 0$, $A_0 = A$, $b_0 = b^{(0)}$, and $W_0 = W$.

- Step 3: If s^k satisfies the optimality condition, return (x^k, y^k, s^k) and **stop**.

- Step 4: Find g^k, the supergradient of $U(s)$ at s^k. Find h^k by solving the following equation
$$A^T Y^0 (S^k)^{-1} A h^k = A^T g^k. \tag{6.25}$$

- Step 5: Set $u^k = (S^k)^{-1} A h^k$ and $W_{k+1} = W_k \cap \{w : \ (u^k)^T (w - w^K) \geq 0\}$. Pick an arbitrary point w^{k+1} from W_{k+1} and find the w^{k+1}-center $(x^{k+1}, y^{k+1}, s^{k+1})$ with respect to $\mathcal{F}$. Set $k = k + 1$ and return to Step 3.

A clear advantage of this algorithm over the one in the s-space is that we do not have to increase the dimension of the w-space at each step and subsequently we do not have to assign weights to the new added constraints. So, the above algorithm is straightforward to implement.

6.3 Modified algorithm in the w-space

In the previous section, we designed a cutting-plane algorithm in the w-space for maximizing the utility function. In this section, we are going to use the properties of the weighted center we derived in Chapter 2 to improve the performance of the algorithm. We introduce two modified versions of the w-space algorithms in this section.

6.3.1 First modified algorithm

As we proved in Chapter 2, for every centric y-vector $\hat{y}$ and any centric s-vector $\hat{s}$, $\hat{w} = \hat{Y}\hat{s}$ is a weight vector in the simplex W. As we are maximizing $U(s)$ over s, roughly speaking, only the s-vector of the weighted center is important for us for each $w \in W$. This is somehow explicit in our algorithm as, for example, the normal to the cutting-plane at each step, given in (6.24), depends on s and y^0 which is the y-vector of the starting point w^0. The algorithm also guarantees to keep $Y^0 s^{opt}$ in the shrunken region at each step. Hence, we lose nothing if we try to work with weight vectors with $y = y^0$.

Consider Lemma 3.2.1 which is about the convex combination of weight vectors. Assume that we have weight vectors w^i, $i \in \{1, \cdots, l\}$, with weighted centers (x^i, y^0, s^i), which means they have the same y-vector. By Lemma 3.2.1, for every set of $\beta_i \in [0,1]$, $i \in \{1,\cdots,l\}$, such that $\sum_{i=1}^l \beta_i = 1$, we have $(\sum_{i=1}^l \beta_i x^i, y^0, \sum_{i=1}^l \beta_i s^i)$ is the w-center where $w := \sum_{i=1}^l \beta_i w^i$. In other words, when the y-vectors are the same, s-vector (equivalently x-vector) of the convex combination of w^i is equal to the convex combination of s^i, $i \in \{1,\cdots,l\}$. This is interesting because if we can update the weight vectors by using the convex combination, we do not need to compute the weighted center. We are going to use this to modify our algorithm.

Assume that the starting point is w^0 with weighted center (x^0, y^0, s^0). The modified algorithm is similar to the algorithm in Section 6.2 and the normal to the cutting-plane is derived by using (6.24). However, in the modified one, all w^i have y-vector equal to y^0. The modified algorithm has two modules:

Module 1: Assume that at Step i, we have $w^i = Y^0 s^i$ and $w^{i-1} = Y^0 s^{i-1}$ with the corresponding normals of the cutting-planes u^i and u^{i-1}. By the choice of w^i, we must have $(u^{i-1})^T(w^i - w^{i-1}) \geq 0$. In the modified algorithm, if we have $(u^i)^T(w^{i-1} - w^i) \geq 0$, then we put $w^{i+1} = (w^i + w^{i-1})/2$ (it is easy to see this weight vector is in the required cut simplex). In this case, we have $y^{i+1} = y^0$ and $s^{i+1} = (s^i + s^{i-1})/2$.

If we have $(u^i)^T(w^{i-1} - w^i) \geq 0$, then the line segment $[w^{i-1}, w^i]$ is no longer in the required cut simplex. However, there exists $t > 0$ such that $\hat{w} := w^i + t(w^i - w^{i-1})$ is in the required cut simplex. We can do a line search to find t and then we set $w^{i+1} = \hat{w}$. In this case, we have $y^{i+1} = y^0$ and $s^{i+1} = s^i + t(s^i - s^{i-1})$.

Module 2: In Module 1, the algorithm always moves along a single line. When the weight vectors in Module 1 get close to each other, we perform Module 2 to get out of that line. To do that, we choose a constant $\epsilon > 0$ and whenever in Module 1 we have $\|w^i - w^{i-1}\|_2 \leq \epsilon$, we perform Module 2. In Module 2, like the algorithm in Section 6.2, we pick an arbitrary weight vector $\hat{w}$ in the remaining cut simplex and compute the weighted center $(\hat{x}, \hat{y}, \hat{s})$. The problem now is that y-vector is not necessarily equal to y^0. However, we said that $\hat{s}$ is important for our algorithm; hence, we consider the weight vector $Y^0\hat{s}$. This new weight vector is not necessarily in the required cut simplex. To solve this problem, we use the same trick as in Module 1. We consider the line containing the line segment $[w^i, Y^0\hat{s}]$ and do a line search to find an appropriate weight vector on this line. To simplify the line search, we consider $(u^i)^T(Y^0\hat{s} - w^i) \geq 0$ and $(u^i)^T(Y^0\hat{s} - w^i) < 0$ separately.

At the end of Module 2, we again come back to Module 1 to continue the algorithm. As can be seen, we only have to find a weighted center in Module 1 which makes the modified algorithm computationally more efficient than the original algorithm, in practice.

6.3.2 Second modified algorithm

Consider the main algorithm and the proof of Proposition 6.2.4. We constructed normal vectors that satisfy (6.23). By using the supergradient inequality, $(g^1)^T(\hat{s} - s^1) \leq 0$ results in $U(\hat{s}) \leq U(s^1)$. Assume that a sequence of s-vectors $\{s^0, s^1, \cdots, s^j\}$ has been created by the algorithm up to iteration j. We may not have access to the value of $U(s_i)$, $i \in \{1, \cdots, j\}$, however, we know that there exists $p \in \{1, \cdots, j\}$ such that $U(s^p) \geq U(s^i)$ for all $i \in \{1, \cdots, j\}$. By the supergradient inequality we must have $(g^i)^T(s^p - s^i) \geq 0$ for all $i \in \{1, \cdots, j\}$ and from (6.23)

$$(u^i)^T(Y^0 s^p - Y^i s^i) = (g^i)^T(s^p - s^i) \geq 0, \quad \forall i \in \{1, \cdots, j\}.$$

This means that $Y^0 s^p$ is a weight vector in the desired cut simplex. Let $\{p_1, \cdots, p_k\}$ be the indices that $(g^i)^T(s^{p_l} - s^i) \geq 0$ for all $i \in \{1, \cdots, j\}, l \in \{1, \cdots, k\}$. By the above explanation, we know that $k \geq 1$ (the s-vector with the highest value so far is in this set.). The idea of the modified algorithm is that when $k > 1$, we put a convex combination of these s-vectors as the new s-vector. We can divide the new algorithm into three modules.

Module 1: $k > 1$: Define $s^{j+1} := \frac{1}{k}(s^{p_1} + \cdots + s^{p_k})$ and $w^{j+1} := Y^0 s^{j+1}$.

Module 2: $k = 1$. We only have one point s^p that $(g^i)^T(s^p - s^i) \geq 0$ for all $i \in \{1, \cdots, j\}$ and by the above explanation we have $U(s^p) \geq U(s^i)$ for all $i \in \{1, \cdots, j\}$. Hence s^p is our best point so far and we use it to find the next one. To do that, we choose a direction ds such that $s^{j+1} = s^p + \alpha ds$. $s^{j+1} - s^p = \alpha ds$ must be in $\mathcal{R}(A)$ and therefore a good choice is the projection of g^p on $\mathcal{R}(A)$. Let us define P_A as the projection matrix to $\mathcal{R}(A)$, then we define $ds = P_A g^p$ and do a line search to find the appropriate α such that $s^{j+1} := s^p + \alpha ds$ is in the desired cut simplex. We also have $w^{j+1} := Y^0 s^{j+1}$.

Module 3: In the first two modules, we do not have to calculate the weighted center. In this module, like the first modified algorithm, when $\|w^{j+1} - w^j\|$ in Module 1 or 2 is smaller than a specified value, we perform an iteration like the original algorithm; pick an arbitrary point inside the cut simplex and compute the weighted center for that.

6.4 Convergence of the algorithm

Introduction of cutting-plane algorithms goes back to 1960's and one of the first appealing ones is the center of gravity version [19]. The center of gravity algorithm has not been used in practice because computing the center of gravity is difficult. However it is worth mentioning for its theoretical convergence. Grünbaum [12] proved that by using any cutting-plane, more than 0.37 of the feasible set is cut out which guarantees a geometric convergence.

The above explanation shows that the convergence of our algorithm depends on the way we choose a weight vector in the cut simplex. Different types of centers, instead of the geometric center, have been proposed in the literature. A group of them use the center of a specific localization set, which is updated at each step. One of the famous of them is the ellipsoid method [18] where the localization set is represented by an ellipsoid containing the optimal point. Ellipsoid method can be related to our algorithm as we can use it find the new weight vectors at each iteration.

The cutting-plane method which is most relevant to our algorithm is the analytic center one, see [23] for a survey on it. In this method, the new point at each iteration is the analytic center of the remaining polyhedron. The complexity of this algorithm has been widely considered in the literature. Nesterov [38] proved the ϵ-accuracy bound of $O(\frac{L^2R^2}{\epsilon^2})$ when the objective function is Lipschitz continuous with constant L, and the optimal set is supposed to lie in ball of diameter R. Goffin, Luo, and Ye [24] considered the feasibility version of the problem and derived a convergence of $O(\frac{n^2}{\epsilon^2})$ calls to the cutting-plane oracle.

Another family of cutting-plane algorithms are based on *volumetric barriers* or *volumetric centers* [29, 30, 26]. Vaidya used the volumetirc center to design a new algorithm for minimizing a convex function over a convex set [29]. More sophisticated algorithms have been developed based on Vaidya's volumetric cutting plane method [30, 26]. It has been proved that the complexity of the volumetric algorithms compares favorably to that of the ellipsoid methods [26].

6.5 Solutions for Practical Concerns

In the previous sections, we introduced an algorithm that is highly cooperative with the DM and proved many interesting features about it. In this section, we set forth some practical concerns about our algorithm and introduce solutions for them.

6.5.1 Driving factors

As we mentioned, one of the main criticisms of classical robust optimization is that it is not practical to ask the DM to specify an m-dimensional ellipsoid for the uncertainty set. Our approach improves this situation by asking much simpler questions. However, as the DM might not be a technical expert, asking the supergradient or gradient of a function in $\mathbb{R}^m$ still has many practical difficulties. In this section, we are going to show how to solve this problem.

The idea is similar to those used in the area of multi-criteria optimization. Consider the system of inequalities $Ax \leq b$ and the corresponding slack vector $s = b - Ax$ representing the problem. What happens in practice is that the DM might prefer to play with a few factors that really matter, we call them *Driving Factors*. For example, the driving factors for a CEO might be budget amount, profit, human resource, etc. We can represent k driving factors by $(c^i)^T x$, $i \in \{1, \cdots, k\}$, and the problem for the DM is to maximize the utility function $U((c^1)^T x, \cdots, (c^k)^T x)$. Similar to the way we added the objective of the linear program to the constraints, we can add k constraints to the problem and write (6.2) as:

$$\begin{aligned} \max \quad & U(s_1, \cdots, s_k) \\ s.t. \quad & s_i = \hat{b}_i - (c^i)^T x \geq 0, \quad i \in \{1, \cdots, k\} \\ & Ax \leq b. \end{aligned} \tag{6.26}$$

As can be seen, the supergradient vector has only k nonzero elements which makes it much easier for the DM to specify it instead of a general vector. (6.2) still has the problem that the DM has to think about a weight vector in a large space. We can solve this by approximating our problem by one in a k-dimensional weight space.

$$\begin{aligned} \max \quad & U(s_1, \cdots, s_k) \\ s.t. \quad & (c^i)^T x \leq \hat{b}_i, \quad i \in \{1, \cdots, k\}. \end{aligned} \tag{6.27}$$

We are actually projecting a high dimensional space to a low dimensional one and therefore losing a lot of information. However, this transformation makes the problem much easier, meaningful and manageable for the DM.

6.5.2 Approximate gradients

In the previous section, we derived a cutting-plane algorithm in the w-space. As can be seen from Propositions 6.2.3 and 6.2.4, for the algorithm we need the supergradients of the utility function $U(s)$. However, we usually do not have an explicit formula for $U(s)$ and our knowledge about it comes from the interaction with the DM. Asking for the supergradient of a function might still be a difficult question from the DM, and it also may be the case that the DM itself does not have an explicit formula for $U(s)$. So we have to simplify our questions from the DM and try to the change our algorithm based on that.

In this section, we try to derive approximate supergradients based on simple questions from the DM. The idea is similar to the one used by Arbel and Oren in [1]. First assume that $U(s)$ is differentiable which means the supergradient at each point is unique and equal to the gradient of the function at that point. Assume that the algorithm is at the point s. By the Taylor's Theorem (first order expansion) for arbitrarily small scalars $\epsilon_i > 0$ we have:

$$\begin{aligned} u_i := U(s + \epsilon_i e_i) \approx U(s) + \frac{\partial U(s)}{\partial s_i} \epsilon_i \\ \Rightarrow \frac{\partial U(s)}{\partial s_i} \approx \frac{u_i - u_0}{\epsilon_i}, \qquad u_0 := U(s). \end{aligned} \tag{6.28}$$

We have $m + 1$ points s and $s + \epsilon_i e_i$, $i \in \{1, \cdots, m\}$ and by the above equation, if we have the value of $U(s)$ at these points, we can find the approximate gradient. But in the absence of true utility function, we have to find these values through proper questions from the DM. Here, we assume that we can ask the DM about the relative preference for the value of the function at these $m+1$ points. For example, DM can use a method called Analytic Hierarchy Process (AHP) to assess relative preference. We use these relative preferences to find the approximate gradient.

Assume that the DM provides us with the priority vector p, then we have the following relationship between p and u_i's

$$\begin{aligned} & \frac{u_i}{u_j} = \frac{p_i}{p_j}, \qquad i, j \in \{0, \cdots, m\}, \\ \Rightarrow \qquad & \frac{u_i - u_0}{u_0} = \frac{p_i - p_0}{p_0}, \\ \Rightarrow \qquad & u_i - u_0 = \beta_0 (p_i - p_0), \qquad \beta_0 := \frac{u_0}{p_0}. \end{aligned} \tag{6.29}$$

Now we can substitute the values of $u_i - u_0$ from (6.29) into (6.28) and we have

$$\nabla U(s) = \beta_0 \begin{bmatrix} \frac{p_1 - p_0}{\epsilon_1} & \cdots & \frac{p_m - p_0}{\epsilon_m} \end{bmatrix}^T. \tag{6.30}$$

The problem here is that we do not have the parameter β_0. However, this parameter is not important in our algorithm because we are looking for normals to our proper hypreplanes and, as it can be seen in (6.21) and (6.24), a scaled gradient vector can also be used to calculate h^0 and h^1. It means we can simply ignore β_0 in our algorithm.

Chapter 7

Illustrative preliminary computational experiments

In this chapter, we present some numerical results to show the performance of the algorithms in the w-space designed in Chapter 6. LP problems we use are chosen from the NETLIB library of LPs. Most of these LP problems are not in the format we have used throughout the paper which is the standard inequality form. Hence, we convert each problem to the standard equality form and then use the dual problem. In this section, the problem $\max\{(c^{(0)})^T x : \ Ax \leq b^{(0)}\}$ is the converted one. In the following, we consider several numerical examples.

Example 1: In this example, we consider a simple problem of maximizing a quadratic function. Consider the ADLITTLE problem (in the converted form) with 139 constraints and 56 variables. We apply the algorithm to function $U_{ij}(s) = -(s_i - s_j)^2$ which makes two slack variables as close as possible. This function may not have any practical application, however, shows a simple example difficult to solve by classical robust optimization.

The stopping criteria is $\|g\| \leq 10^{-6}$. For U_{23} the algorithm takes 36 iterations and returns $U_{23} = -5 \times 10^{-11}$. For U_{34} the algorithm takes 35 iterations and returns $U_{34} = -2.4 \times 10^{-12}$.

Example 2: Consider the ADLITTLE problem and assume that three constraints $\{68, 71, 74\}$ are important for the DM. Assume that the DM estimates that there is 20 percent uncertainty in the RHS of these inequalities. We have $(b_{68}, b_{71}, b_{74}) = (500, 493, 506)$ and so the desired slack variables are around $(s_{68}, s_{71}, s_{74}) = (100, 98, 101)$. By using the classical robust optimization method that satisfies the worst case scenario, the optimal objective value is $obj_c = 1.6894 \times 10^5$.

Now assume that the DM takes the following utility function:

$$U_1(s) = t_{68} \ln(s_{68}) + t_{71} \ln(s_{71}) + t_{74} \ln(s_{74}) + t_m \ln(s_m).$$

This function is a NDAS function that we defined in Definition 6.2.1. Assume that the DM set $t_{68} = t_{71} = t_{74} = 1$ and $t_m = 10$. By using our algorithm, we get the objective value of $obj_1 = 1.7137 \times 10^5$ with the slack variables $(s_{68}, s_{71}, s_{74}) = (82, 83, 132)$. As we observe, the objective value is higher than the classical robust optimization method while two of the slack conditions are not satisfied. However, the slack variables are close to the desired ones. If the DM sets $t_m = 20$, we get the objective value of $obj_2 = 1.9694 \times 10^5$ with the slack variables $(s_{68}, s_{71}, s_{74}) = (40, 41, 79)$. However, all the iterations might be interesting for the DM. The following results are also returned by the algorithm before the optimal one:

$$obj_3 = 1.8847 \times 10^5, \quad (s_{68}, s_{71}, s_{74}) = (56, 58, 83),$$
$$obj_4 = 1.7 \times 10^5, \quad (s_{68}, s_{71}, s_{74}) = (82, 84, 125).$$

Now assume that the DM wants to put more weight on constraints 68 and 71 and so set $t_{68} = t_{71} = 2, t_{74} = 1$ and $t_m = 20$. In this case, the algorithm returns $obj_5 = 1.8026 \times 10^5$ with the slack variables $(s_{68}, s_{71}, s_{74}) = (82, 84, 64)$.

Example 3: In this example, we consider the DEGEN2 problem (in the converted form) with 757 constraints and 442 variables. The optimal solution of this LP is $obj_1 = -1.4352 \times 10^3$. Assume that constraints 245, 246, and 247 are important for the DM who wants them as large as possible, however, at the optimal solution we have $s(245) = s(246) = s(247) = 0$. The DM also wants the optimal objective value to be at least -1.5×10^3. As we stated before, we add the objective function as a constraint to the system. To have the objective value at least -1.5×10^3, we can add this constraint as $c^T x = -1500 + s_{m+1}$. For the utility function, the DM can use the NDAS function

$$U(s) = \ln(s_{245}) + \ln(s_{246}) + \ln(s_{247}).$$

By running the algorithm for the above utility function, we get $(s_{245}, s_{246}, s_{247}) = (7.75, 17.31, 17.8)$ with objective value $obj_2 \approx -1500$ after 50 iterations and $(s_{245}, s_{246}, s_{247}) = (15.6, 27.58, 27.58)$ with $obj_3 \approx -1500$ after 100 iterations.

Example 4: We put a stopping criteria in the algorithm on the norm of the supergradient. The DM can also have a stopping criteria maybe because of being satisfied or getting tired. In this example, we consider the SCORPION problem (in the converted form) with 466 constraints and 358 variables. The optimal objective value of this LP is $obj_1 = 1.8781 \times 10^3$. We consider the following two NDAS utility functions:

$$U_1(s) = \ln(s_{m+1}) + \sum_{i=265}^{274} \ln(s_i),$$
$$U_2(s) = 5\ln(s_{m+1}) + \sum_{i=265}^{274} \ln(s_i). \tag{7.1}$$

In this example, we apply the original and the 2nd modified algorithms to both $U_1(s)$ and $U_2(s)$. The improvement in the the utility function value after 200 iterations is shown in Fig 7.1. As can be seen, the rate of increase in the utility function is decreasing after

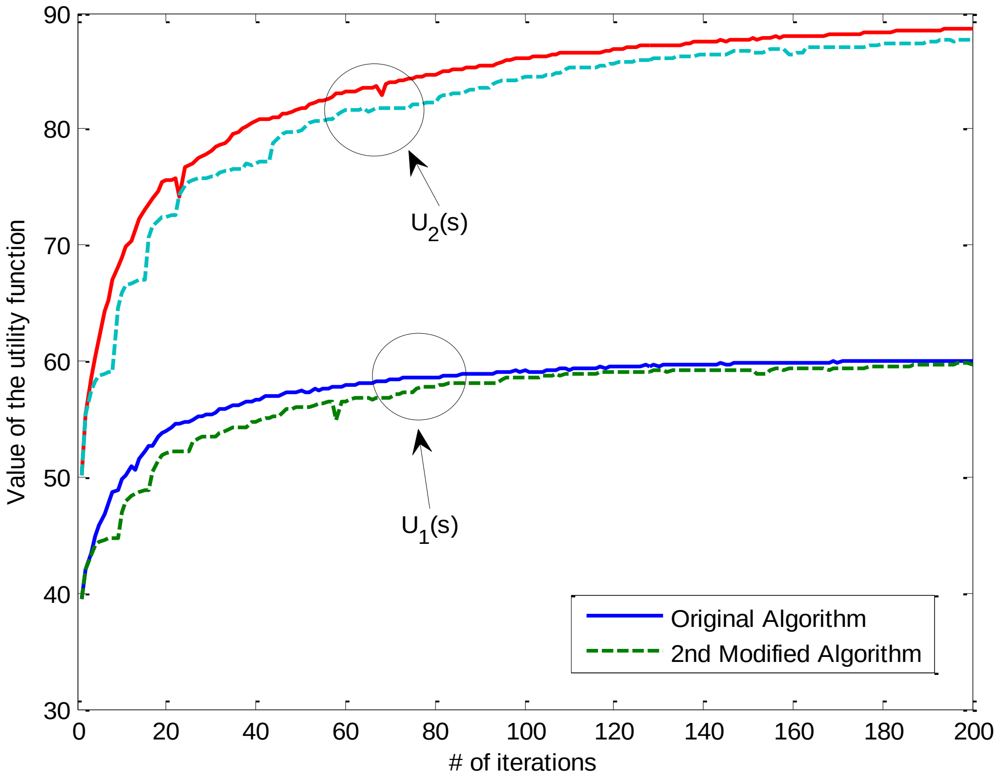

Figure 7.1: Value of the utility function versus the number of iterations.

each iteration. For this problem, the algorithm does not stop by itself and continues until the satisfaction of the DM. The DM can stop the algorithm, for example, when the rate of increase is less than a specified coefficient. The other point is that, for this example, the rate of improvement for the 2nd modified algorithm is almost as good as the original one. However, in the modified algorithm, the weighted center is computed around 40 times during the 200 iterations which is much less computational work.

Example 5: In this example, we are going to consider utility functions introduced at the end of Chapter 5. Consider problem SCORPION with optimal objective value of $obj_1 = 1.8781 \times 10^3$. Assume that the uncertainty in constraints 211 to 215 are important for the DM and we have $\|\Delta b_i\|_1 = 0.7b_i^{(0)}$, $i \in \{211, \cdots, 215\}$, where Δb_i was defined in (3.5). Let $\hat{x}$ be the solution of MATLAB's LP solver, then we have $s_{211} = \cdots = s_{215} = 0$ which is not satisfactory for the DM. Besides that, assume that the DM wants the objective value to be at least 1800. To satisfy that, we add the $(m+1)$th constraint as $s_{m+1} = -1800 + (c^{(0)})^T x$ which guarantees $(c^{(0)})^T x \geq 1800$. For the utility function, first we define $u_i(s_i)$, $i \in \{211, \cdots, 215\}$ similar to Fig 5.2 with $\epsilon_i^1 = \|\Delta b_i\|_1 = 0.7 b_i^{(0)}$ and $\epsilon_i^2 = \infty$. So we have for $i \in \{211, \cdots, 215\}$:

$$u_i(s_i) = \begin{cases} s_i & s_i < \|\Delta b_i\|_1 \\ \|\Delta b_i\|_1 & s_i \geq \|\Delta b_i\|_1. \end{cases} \tag{7.2}$$

Now, we can define $U(s) := \sum_{i=211}^{215} \ln u_i(s_i)$. By running the algorithm, the supergradient goes to zero after 65 iterations and the algorithm stops. Denote the solution by x^*, then the results are as follows:

$$\begin{aligned} &(c^{(0)})^T x^* = 1800.3, \\ &b_{211}^{(0)} = 3.86, \quad b_{212}^{(0)} = 48.26, \quad b_{211}^{(0)} = 21.81, \quad b_{211}^{(0)} = 48.26, \quad b_{211}^{(0)} = 3.86, \\ &s_{211}^* = 3.29, \quad s_{212}^* = 19.47, \quad s_{211}^* = 7.39, \quad s_{211}^* = 16.97, \quad s_{211}^* = 3.24. \end{aligned} \tag{7.3}$$

Now, assume that the DM wants the objective value to be at least 1850 and the $(m+1)$th constraint becomes $s_{m+1} = -1850 + (c^{(0)})^T x$. It this case, the norm of the supergradient reaches to zero after 104 iterations. The norm of supergradients versus the number of iterations is shown in Fig 7.2 for these two cases. Denote the solution after 100 iterations by $\bar{x}^*$, then we have:

$$\begin{aligned} &(c^{(0)})^T \bar{x}^* = 1850, \\ &\bar{s}_{211}^* = 1.22, \quad \bar{s}_{212}^* = 16.74, \quad \bar{s}_{211}^* = 6.80, \quad \bar{s}_{211}^* = 14.54, \quad \bar{s}_{211}^* = 1.25. \end{aligned} \tag{7.4}$$

Let $\bar{x}$ be the returned value in the second case after 65 iterations. It is clearly not robust feasible, however, we can use bound (5.3) to find an upper bound on the probability of infeasibility. Assume that $N = 10$ and all the entries of Δb_i are equal. Then bound (5.3) reduces to $B(N, \delta_i N)$ where $\delta_i = \frac{s_i}{\|\Delta b_i\|_1}$. The probability of infeasibility of $\bar{x}$ for constraints 211 to 215 is in Table 7.1 by using bound (5.3):

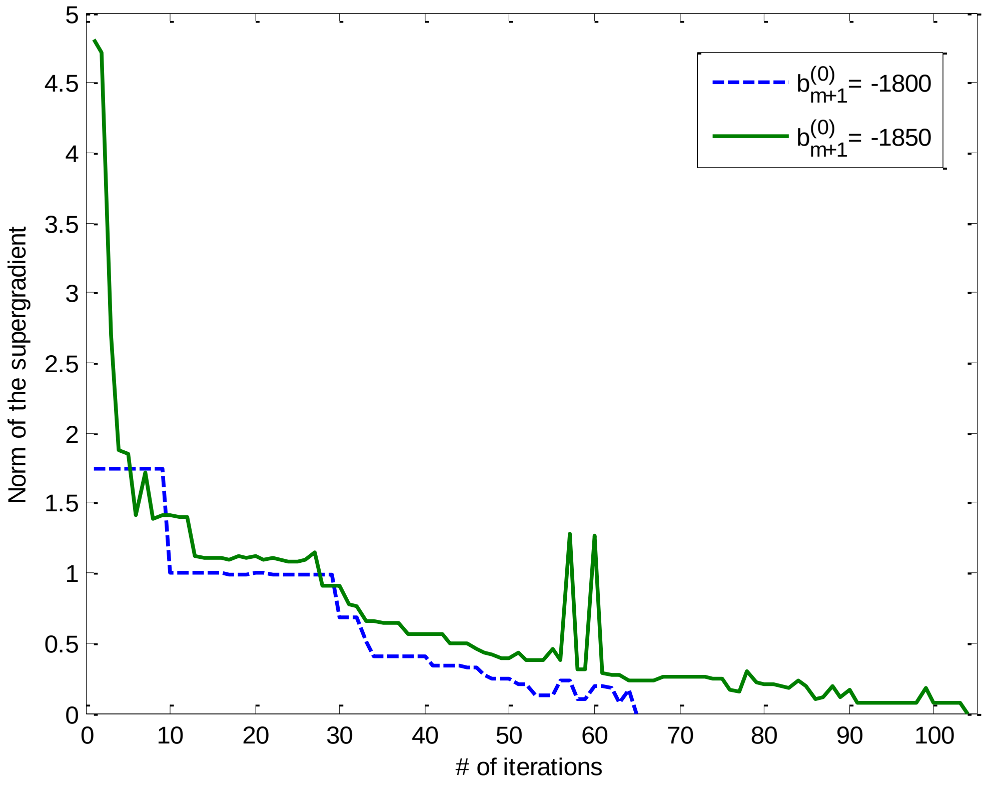

Figure 7.2: Norm of the supergradient versus the number of iterations for Example 5.

Table 7.1: The probability of infeasibility of $\bar{x}$ for constraints 211 to 215.

i	$\Pr(\langle a_j, \bar{x}\rangle > \tilde{b}_j)$
211	0
212	0.0827
213	0.0018
214	0.0866
215	0

Chapter 8

Extensions and conclusion

8.1 Extension to Semidefinite Optimization (SDP)

Semidefinite programming is a special case of Conic programming that the cone is a semidefinite cone or the direct product of semidefinite cones. Many convex optimization problems can be modeled by SDP which attracted a lot of interest recently. Since our method is based on a barrier function for a polytope in $\mathbb{R}^n$, it can be generalized and used as an approximation method for robust semidefinite programming that is NP-hard for ellipsoidal uncertainty sets. An SDP problem can be formulated as follows

$$\begin{aligned} \sup \quad & \langle \tilde{c}, x \rangle, \\ \text{s.t.} \quad & \sum_{j=1}^{t_i} A_i^{(j)} x_j + S_i = \tilde{B}_i, \quad \forall i \in \{1,2,...,m\}, \\ & S_i \succeq 0, \quad \forall i \in \{1,2,...,m\}, \end{aligned}$$

where $A_i^{(j)}$ and $\tilde{B}_i$ are symmetric matrices of appropriate size, and $\succeq$ is the generalized inequality; for two square matrices C_1 and C_2 with the same size, we have $C_1 \succeq C_2$ iff $C_1 - C_2$ is a semidefinite matrix. For every $i \in \{1, \cdots, m\}$, define

$$\mathcal{F}_i := \{x \in \mathbb{R}^n : \sum_{j=1}^{t_i} A_i^{(j)} x_j \preceq \tilde{B}_i\}$$

Assume that $\text{int}(\mathcal{F}_i) \neq \emptyset$ and let $F_i : \text{int}(\mathcal{F}_i) \to \mathcal{R}$ be a self-concordant barrier for $\mathcal{F}_i$. The typical self-concordant barrier for SDP is $F_i(x) = -\log\det(\tilde{B}_i - \sum_{j=1}^{t_i} A_i^{(j)} x_j)$. Assume

$$\mathcal{F} := \bigcap_{i=1}^{m} \mathcal{F}_i$$

is bounded and its interior is nonempty. Now like the definition of the weighted center for LP, we can define a weighted center for SDP. For every $w \in \mathbb{R}^m_{++}$, we can define the weighted center as follows:

$$\arg\min \left\{ \sum_{i=1}^{m} w_i F_i(x) : x \in \mathcal{F} \right\} \tag{8.1}$$

The problem with this definition is that we do not have many of the interesting properties we proved for LP. The main one is that the weighted centers do not cover the whole feasible region and we cannot sweep the whole feasible region by moving in the w-space. There are other notions of weighted centers that solve this problem; however, they are more difficult to work with. Extending the results we derived for LP to SDP can be a good research to follow.

8.2 Quasi-concave utility functions

The definition of the quasi-concave function is as follows:

Definition 8.2.1. *A function $f : \mathbb{R}^n \to \mathbb{R}$ is* ***quasi-concave*** *if its domain is convex, and for every $\alpha \in \mathbb{R}$, the set*

$$\{x \in \text{dom} f \ : \ f(x) \geq \alpha\}$$

is also convex.

All concave functions are quasi-concave, however, the converse is not true. Quasi-concave functions are important in many fields such as game theory and economics. In microeconomics, many utility functions are modeled as quasi-concave functions. For differentiable functions, we have the following useful proposition:

Proposition 8.2.1. *A differentiable function f is quasi-concave if and only if the domain of f is convex and for every x and y in* dom$f$ *we have:*

$$f(y) \geq f(x) \quad \Rightarrow \quad (\nabla f(x))^T (y - x) \geq 0 \tag{8.2}$$

(8.2) is similar to (6.3), which is the property of the supergradient we used to design our algorithms. The whole point is that for a differentiable quasi-concave function $U(s)$ and any arbitrary point s^0, the maximizers of $U(s)$ are in the half-space $(\nabla U(s^0))^T (s - s^0) \geq 0$. This means that we can extend our algorithms to differentiable quasi-concave utility functions simply by replacing supergradient with gradient, and all the results for s-space and w-space stay valid.

8.3 Conclusion

In this thesis, we presented new algorithms in a framework for robust optimization designed to mitigate some of the major drawbacks of robust optimization in practice. Our algorithms have the potential of increasing the applicability of robust optimization. Some of the advantages of our new algorithms are:

1. Instead of a single, isolated, and very demanding interaction with the DM, our algorithm interacts continuously with the DM throughout the optimization process with much more reasonable demands from the DM in each iteration. One of the benefits of our approach is that the DM "learns" what is feasible to achieve throughout the process. Another benefit is that the DM is more likely to be satisfied with the final solution. Moreover, since the DM is personally involved in the production of the final solution, s/he bears some responsibility for it and is more likely to adapt it in practice.

2. Some of our algorithms operate in the weight space. This helps reduce the dimension of the problem, simplify the demands on the DM while computing the most important aspect of the problem at hand.

3. Weight space and weighted-analytic-centers approach embeds a "highly differentiable" structure into the algorithms. Such tools are extremely useful in both the theory and applications of optimization. In contrast, classical robust optimization and other competing techniques usually end up delivering a final solution where differentiability cannot be expected.

Developing similar algorithms for semidefinite programming can be a future research topic. As we explained in Section 8.1, we can define a similar notion of weighted center for SDP. However, these weighted centers do not have many properties we used for LP, and we may have to switch to other notions of weighted centers that are more difficult to work with.

References

[1] A. Ardel, and S. Oren, Using approximate gradients in developing an interactive interior primal-dual multiobjective linear programming algorithm, *European Journal of Operational Research*, 89 (1996), 202–211.

[2] A. Ben-Tal, L. El Ghaoui, and A. Nemirovski, *Robust Optimization*, Princeton Series in Applied Mathematics, (2009).

[3] A. Ben-Tal and A. Nemirovski, Robust Solutions Of Uncertain Linear Programs, *Operation Research Letters*, 25 (1999) 1–13.

[4] A. Ben-Tal and A. Nemirovski, Robust Solutions Of Linear Programming Problems Contaminated With Uncertain Data, *Math. Prog.* 88 (2000) 411–424

[5] A. Ben-Tal and A. Nemirovski, Robust Convex Optimization, *Mathematics of Operations Research*, 23 (1998) 769–805.

[6] A. Ben-Tal and A. Goryashko and E. Guslitzer and A. Nemirovski, Adjustable Robust Solutions Of Uncertain Linear Programs, *Mathematical Programming*, 99 (2004) 351–376.

[7] A. Ben-Tal, S. Boyd and A. Nemirovski, Extending Scope Of Robust Optimization: Comprehensive Robust Counterparts of Uncertain Problems, *Mathematical Programming*, 107 (2006) 63–89.

[8] A. Ghaffari Hadigheh and T. Terlaky, Generalized Support Set Invariancy Sensitivity Analysis In Linear Optimization, *Journal of Industrial and Management Optimization*, 2 (2006) 1–18.

[9] A. Ghaffari Hadigheh and T. Terlaky, Sensitivity Analysis In Linear Optimization: Invariant Support Set Intervals, *European Journal of Operational Research*, 169 (2006) 1158-1175.

[10] A. L. Soyster, Convex programming with set-inclusive constraints and applications to inexact linear programming, *Operations Research*, 21 (1973), 1154-1157.

[11] A. Nemirovski, and A. Shapiro, Convex approximations of chance constrained programs, *SIAM Journal of Optimization*, 4 (2006) 969–996.

[12] B. Grünbaum, Partitions of mass-distributions and convex bodies by hyperplanes, *Pacific J. Math.*, 10 (1960), 1257-1261.

[13] D. Bertsimas and D. Pachamanova and M. Sim, Robust Linear Optimization Under General Norms, *Operations Research Letters*, 32 (2004) 510–516.

[14] D. Bertsimas, and I. Popescu, Optimal Inequalities in Probability Theory- a Convex Optimization Approach, *SIAM J. OPTIM*, 15 (2005), no. 3, 780–804.

[15] D. Bertsimas and M. Sim, Robust Discrete Optimization And Network Flows, *Math. Program.*, 98 (2003)49–71.

[16] D. Bertsimas and M. Sim, The Price Of Robustness, *Operations Research*, 52 (2004) 35–53.

[17] D. Bertsimas and M. Sim, Tractable Approximations To Robust Conic Optimization Problems, *Mathematical Programming*, June 2005.

[18] D. B. Yudin and A.S. Nemirovski, Informational complexity and efficient methods for solving complex extremal problems, *Matekon*, 13 (1977), 25-45.

[19] D. J. Newman, Location of the maximum on unimodal surfaces, *Journal of the ACM*, 12 (1965), 395-398.

[20] E. Erdoğan and G. Iyengar, Ambiguous Chance Constrained Problems And Robust Optimization, *Mathematical Programming*, 107 (2006) 37–90.

[21] F. Ordonez and J. Zhao, Robust Capacity Expansion Of Network Flows, USC-ISE Working paper 2004–01.

[22] J. H. Gallier, *Geometric methods and applications: for computer science and engineering*, Springer, 2001.

[23] J. L. Goffin and J. P. Vial, Convex non-differentiable optimization: a survey focused on the analytic center cutting-plane method, *Optimization Methods & Software*, 17 (2002), 805-867.

[24] J. L. Goffin, Z. Q. Luo, and Y. Ye. On the complexity of a column generation algorithm for convex and quasiconvex feasibility problems. *Large Scale Optimization: State of the Art, Kluwer Academic Publishers*, (1993). 187–196.

[25] J. M. Mulvey, R. J. Vanderbei, and S. A. Zenios, Robust Optimization of Large-Scale Systems, *Operations Research*, 43 (1995), 264–281.

[26] K. M. Anstreicher, On Vaidya's Volumetric Cutting Plane Method for Convex Programming, *Mathematics of Operations Research* , 22 (1997), 63–89.

[27] L. B. Miller, and H. Wagner, Chance-constrained programming with joint constraints, *Operations Research*, 13 (1965), 930–945.

[28] L. El. Ghaoui and F. Oustry and H. Lebret, Robust Solutions To Uncertain Semidefinite Programs *SIAM J. OPTIM* 9 (1998) 33–52.

[29] P. M. Vaidya, A new algorithm for minimizing convex functions over convex sets, *Symposium on Foundations of Computer Science*, (1989), 338–343.

[30] P. M. Vaidya, and D. S. Atkinson, A Technique for Bounding the Number of Iterations in Path Following Algorithms, *Complexity in Numerical Optimization*, World Scientific, Singapore, (1993), 462–489.

[31] R. T. Rockafellar, *Convex Analysis*, Princeton University Press, 1997.

[32] S. Boyd, and L. Vanderberghe, *Convex Optimization*, Cambridge University Press, 2004.

[33] S. Mudchanatongsuk, F. Ordonez, and and J. Liu, Robust Solutions For Network Design Under Transportation Cost And Demand Uncertainty, USC ISE Working paper 2005–05.

[34] S. Moazeni, *Flexible Robustness in Linear Optimization*, Master's Thesis, University of Waterloo, 2006.

[35] T. Santoso, S. Ahmed, M. Goetschalckx, and A. Shapiro, A stochastic programming approach for supply chain network design under uncertainty, *European Journal of Operational Research*, 167 (2005) 96-115.

[36] V. S. Iyengar, J. Lee, and M. Campbell, Q-Eval: Evaluating Multiple Attribute Items Using Queries, *Proceedings of the 3rd ACM conference on Electronic Commerce*, (2001) 144-153.

[37] W. Hoeffding, Probability Inequalities For Sums Of Bounded Random Variables, *Journal of the American Statistical Association* 58 (1963) 13-30.

[38] Yu. Nesterov. Complexity estimates of some cutting-plane methods based on the analytic barrier. *Mathematical Programming, Series B*, 69 (1995), 149-176.

[39] Yu. Nesterov, and A. Nemirovskii, *Interior Point Polynomial Algorithm in Convex Programming*, SIAM; Studies in Applied and Numerical Mathematics, 1994.