Robust Linear Optimization with Recourse: Solution Methods and Other Properties

by

Tara L. Terry

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Industrial and Operations Engineering)
in The University of Michigan
2009

Doctoral Committee:

Associate Professor Marina A. Epelman, Chair
Professor Romesh Saigal
Assistant Professor Amitabh Sinha
Assistant Professor Aurelie Thiele, Lehigh University

ACKNOWLEDGEMENTS

I would like to thank my advisor, Marina A. Epelman, for all of her support, guidance, and assistance throughout my time in graduate school. I have learned a great deal from her, more than I can express here, and I am grateful for the commitment she made to me in helping me develop myself as a researcher, an educator, and a person.

I am appreciative of my dissertation committee's willingness to perform their duties; their comments and critiques were informative and only improved my research and results. It was helpful to see my research from another perspective, which helped me grow as a researcher.

I also want to thank the IOE department, both for the financial support through my graduate student instructor position and fellowship and the caring that was displayed by the faculty, staff, and students in the department. The people in the department were so generous with their time and wisdom and I am saddened to leave such a great place.

I also want to thank the National Science Foundation and the Elizabeth C. Crosby Research Award. This work was partially supported by the National Science Foundation under grant CCF-0306240.

Last, but not least, I want to thank my family and, in particular, Serge. Their love and support helped me to remain strong, to keep working towards my goals, and to look positively at setbacks that ultimately helped me grow, learn, and succeed. I am indebted to Serge for his enthusiasm for learning and educating through classwork and research. I didn't believe I was Ph.D. material, but the strength of his belief in me and of my future success has been proven correct by the very existence of this dissertation. He introduced me to the academia world just at the right time and my life has been forever altered and for that I am grateful.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

Table

CHAPTER I

Introduction

The unifying theme of this dissertation is robust optimization; the study of solving certain types of convex robust optimization problems and the study of bounds on the distance to infeasibility for certain types of robust optimization problems. Robust optimization has recently emerged as a new modeling paradigm designed to address data uncertainty in mathematical programming problems by finding an optimal solution for the worst-case instances of unknown, but bounded, parameters. Parameters in practical problems are not known exactly for many reasons: measurement errors, round-off computational errors, even forecasting errors, which creates a need for a robust approach. The advantages of robust optimization are two-fold: guaranteed feasible solutions against the considered data instances and not requiring the exact knowledge of the underlying probability distribution, which are limitations of chance-constraint and stochastic programming. Adjustable robust optimization, an extension of robust optimization, aims to solve mathematical programming problems where the data is uncertain and sets of decisions can be made at different points in time, thus producing solutions that are less conservative in nature than those produced by robust optimization.

This dissertation has two main contributions: presenting a cutting-plane method for solving convex adjustable robust optimization problems and providing preliminary results for determining the relationship between the conditioning of a robust linear program under structured transformations and the conditioning of the equivalent second-order cone program under structured perturbations. The proposed algorithm is based on Kelley's method and is discussed in two contexts: a general convex optimization problem and a robust linear optimization problem with recourse under right-hand side uncertainty. The proposed algorithm is then tested on two different robust linear optimization problems with recourse: a newsvendor problem with simple recourse and a production planning problem with general recourse, both under right-hand side uncertainty. Computational results and analyses are provided. Lastly, we provide bounds on the distance to infeasibility for a second-order cone program that is equivalent to a robust counterpart under ellipsoidal uncertainty in terms of quantities involving the data defining the ellipsoid in the robust counterpart.

In this chapter, we first provide an overview of recent developments in the robust optimization literature and then discuss the contributions of this dissertation.

1.1 Literature Review

The concept of robust feasibility was pioneered in 1973 by Soyster [61], who proposed a model that guarantees feasibility for all instances of the parameters within a convex set, but the convex set is defined via set containment instead of the usual set of convex inequalities. It wasn't until the mid-1990s that robust feasibility saw a renewed interest. Ben-Tal and Nemirovski [7] hint at robust optimization, but hadn't yet coined the phrase, in a study of robust truss topology design and model the problem as a semidefinite program. Ben-Tal and Nemirovski [8] performed a comprehensive analysis detailing the solvability of various convex robust optimization problems for ellipsoidal uncertainty sets and intersections of ellipsoidal uncertainty sets under the title robust convex optimization. Further work of Ben-Tal and Nemirovski includes uncertain linear programs [9, 10], uncertain quadratic and conic quadratic programs written in conjunction with Roos [12], and uncertain semidefinite and conic quadratic programs [11]. Independently of Ben-Tal and Nemirovski's truss topology work, in [28] El Ghaoui and Lebret studied least-squares problems with ellipsoidal uncertainty, which can be formulated as semidefinite programs, and resulted in the further analysis of semidefinite programs by El Ghaoui et al. in [29]. Recently the robust optimization approach has been considered for portfolio selection problems (Goldfarb and Iyengar [41]), integer programming and network flows (Bertsimas and Sim [14]), supply chain management (Bertsimas and Thiele [17]), inventory theory (Berstimas and Thiele [18]), radiation treatment planning (Chu et al. [26]), and many other applications.

Robust counterparts (RCs) are often semi-infinite optimization problems which do not immediately lend themselves to efficient solution methods, such as interior-point methods. One solution method is to express the robust counterpart as an explicit optimization problem which can then be solved using efficient techniques; e.g., Ben-Tal and Nemirovski [8] show that the robust counterpart of an uncertain convex quadratically constrained quadratic program (QCQP) with ellipsoidal uncertainty can be reformulated as a semidefinite program. In contrast, instead of focusing on ellipsoidal uncertainty, Goldfarb and Iyengar [40] investigate which uncertainty sets allow you to reformulate the convex QCQP as a second-order cone program and give examples of when these uncertainty sets would arise naturally. The solution method of reformulating a RC as an explicit optimization problem usually leads to an increase in complexity, which could lead to intractability. For computationally intractable RCs with specific uncertainty sets, computationally tractable approximate RCs were given for conic quadratic problems (Ben-Tal et al. [12]), for semidefinite problems (Ben-Tal and Nemirovski [11]), and for robust conic quadratic optimization problems (Bertsimas and Sim [16]). In spirit of the above reformulation work, Averbakh and Zhao [3] reformulate RCs for a general class of mathematical programming problems where the uncertainty set is represented by a system of convex inequalities, allowing their work to be applicable to a wider range of problems and more complicated uncertainty sets. Bertsimas and Sim [15] study polyhedral uncertainty, which does not increase the complexity of the problem at hand, and explicitly quantify the trade-off between performance and conservatism (introducing what is now called a budget of uncertainty) in terms of probabilistic bounds of constraint violation. A highlight of their approach is that it can be easily extended to discrete optimization (Bertsimas and Sim[14]). Bertsimas and Thiele [18] use the above mentioned Bertsimas and Sim framework to address uncertainty on the underlying distributions in a multi-period inventory problem, showcasing the potential of robust optimization for dynamic decision-making in the presence of randomness.

Robust optimization has a modeling disadvantage: having to make every decision before seeing the realization of the data, thus producing overly conservative solutions. There are many optimization problems in which only a subset of the decisions must be made before the realization of the data, but the remaining decisions can be made after observing the realized data. Multi-period production planning problems represent a class of problems for which this separation of variables into groups of decisions to be made at different points in time occurs naturally. The case when groups of decisions can be made at two points in time can be modeled via a two-stage formulation called the adjustable robust counterpart (ARC) where the second-stage decisions are referred to as the recourse action. ARCs are very similar to two-stage stochastic programs, but the solution methods differ as stochastic programming requires some knowledge of the underlying probability distribution while ARCs require a known uncertainty set.

The greater flexibility of the ARC results in an additional increase in complexity on top of that of the RC, and frequently leads to computationally intractable problems. It has been shown by Ben-Tal et al. [6] that the ARC of an LP is computationally tractable, in fact equivalent to a larger LP, if the uncertainty set is given as a convex hull of a finite number of points and the recourse coefficient matrix is fixed. When either of these conditions fails, the ARC can be computationally intractable, which leads them to restrict the second-stage variables to affine functions of the data. In [1], Atamtürk and Zhang model a network flow and design under uncertain demand using a two-stage optimization model that does not involve affinely adjustable decision variables. Ordóñez and Zhao in [51] present a tractable ARC for transportation networks for a multi-commodity flow problem with a single source and sink per commodity and uncertain demand and travel time represented by bounded convex sets. When the underlying problem is nonlinear, Takeda et al. in [62] show that for problems with polytopic uncertainty, quasi-convexity of the optimal value function of certain subproblems involving maximization over the uncertainty set is sufficient for reducing the ARC to an explicit optimization problem.

Another solution method for solving ARCs takes an iterative approach and looks to use cutting-plane algorithms. Bienstock and Özbay [21] and Bienstock [20] use Bender's decomposition (delayed-constraint generation) to solve robust optimization problems for specific applications: determining a robust basestock level under uncertain demands and robust portfolio optimization with uncertain returns, respectively.

Their approach alternates between solving a restricted master problem that includes a limited subset of possible data realizations to determine an approximate solution and an adversarial problem which finds the worst-case data realization for the approximate solution found. The newly identified data instance is then added to the restricted master problem and the process is repeated. While writing (Thiele et al.) [63], which contains, but is not limited to, work found in Chapters II, III, and IV of this dissertation, we became aware of the recent work by Mutapcic and Boyd [47], which applies cutting-plane methods to convex robust optimization problems. The overall idea is the same: applying cutting-plane methods, in particular Kelley's method, to convex robust optimization problems. However, convex adjustable robust optimization problems (or robust problems with recourse) considered in Chapters II, III, and IV present additional challenges (specifically in solving the adversarial problem) and a significant portion of the work is devoted to the discussion of solution methods of adversarial problems arising in problems with recourse under right-hand side uncertainty.

1.2 Contributions

We address right-hand side uncertainty in linear programming problems with recourse by modeling random variables as uncertain parameters in a polyhedral uncertainty set. The level of conservatism of the optimal solution is flexibly adjusted by setting a parameter called the "budget of uncertainty" to an appropriate value. A cutting-plane solution method, based on Kelley's method, is presented for solving adjustable robust linear programs. This method is similar to, but less computationally demanding than, Benders' decomposition. We provide techniques for finding the worst-case realizations of the uncertain parameters within the polyhedral uncertainty set for problems with simple and general recourse and provide computational experiments and analysis for both. Lastly, we propose several data transformations for a robust counterpart with ellipsoidal uncertainty and then bound the distance to infeasibility of the equivalent second-order cone program by quantities involving the data defining the ellipsoid.

The structure of this dissertation is as follows. Chapter II discusses sufficient conditions for convexity of the general ARC, cutting-plane algorithms for a general convex program, and the details and difficulties of computing subgradients and objective function values. Chapter III contains an analysis of robust linear programs with simple recourse, computational results for solving a newsvendor problem with simple recourse with Kelley's method, and experimental results that motivated the choice of Kelley's method over other cutting-plane algorithms. Chapter IV presents an analysis for robust linear programs with general recourse and computational results for solving a production planning problem with general recourse with Kelley's method. Chapter V proposes several transformations to the robust counterpart data, provides a definition for distance to infeasibility for each type of transformation, and then bounds the distance to infeasibility of the equivalent second-order cone program by quantities involving the data defining the ellipsoid. Chapter VI will provide a conclusion and discuss directions for future work.

CHAPTER II

Cutting-Plane Algorithms for Solving Adjustable Robust Optimization Problems

2.1 Problem Overview

2.1.1 Optimization With Recourse

The focus of Chapters II, III, and IV is on two-stage linear optimization with right-hand side uncertainty [1], which was first described by Dantzig in [27]. The deterministic problem can be formulated as:

$$\begin{aligned} \min \quad & \mathbf{c}^T\mathbf{x} + \mathbf{d}^T\mathbf{y} \\ \text{s.t.} \quad & \mathbf{A}\ \mathbf{x} + \mathbf{B}\ \mathbf{y} = \mathbf{b}, \\ & \mathbf{x}, \mathbf{y} \geq \mathbf{0}, \end{aligned} \tag{2.1}$$

with the following notations:

- $\mathbf{x}$: the first-stage decision variables,
- $\mathbf{y}$: the second-stage decision variables,
- $\mathbf{c}$: the first-stage costs,
- $\mathbf{d}$: the second-stage costs,
- $\mathbf{A}$: the first-stage coefficient matrix,
- $\mathbf{B}$: the second-stage coefficient matrix,
- $\mathbf{b}$: the requirement vector.

[1] Most of the material discussed in this chapter can also be found in Thiele et al. [63].

In many applications, the requirement vector is random and the decision-maker implements the first-stage ("here-and-now") variables without knowing the actual requirements, but chooses the second-stage ("wait-and-see") variables only after the uncertainty has been revealed. This has traditionally been modeled using stochastic programming techniques, i.e., by assuming that the requirements obey a known probability distribution and minimizing the expected cost of the problem. In mathematical terms, we define the recourse function, once the first-stage decisions have been implemented and the realization of the uncertainty is known, as:

$$
\begin{aligned}
Q(\mathbf{x},\mathbf{b}) = \quad & \min \quad \mathbf{d}^T\mathbf{y} \\
& \text{s.t.} \quad \mathbf{B}\ \mathbf{y} = \mathbf{b} - \mathbf{A}\ \mathbf{x}, \\
& \qquad\ \ \mathbf{y} \geq \mathbf{0},
\end{aligned}
\tag{2.2}
$$

and the stochastic counterpart of problem (2.1) can be formulated as a nonlinear problem:

$$
\begin{aligned}
\min \quad & \mathbf{c}^T\mathbf{x} + \mathbf{E}_{\mathbf{b}}[Q(\mathbf{x},\mathbf{b})] \\
\text{s.t.} \quad & \mathbf{x} \geq \mathbf{0}.
\end{aligned}
\tag{2.3}
$$

If the uncertainty is discrete, consisting of Ω possible requirement vectors each occurring with probability π_ω, $\omega = 1, \ldots, \Omega$, problem (2.3) becomes a linear programming problem:

$$
\begin{aligned}
\min \quad & \mathbf{c}^T\mathbf{x} + \sum_{\omega=1}^{\Omega} \pi_\omega \cdot \mathbf{d}^T\mathbf{y}_\omega \\
\text{s.t.} \quad & \mathbf{A}\ \mathbf{x} + \mathbf{B}\ \mathbf{y}_\omega = \mathbf{b}_\omega, \qquad \forall\omega, \\
& \mathbf{x},\ \mathbf{y}_\omega \geq \mathbf{0}, \qquad\qquad \forall\omega.
\end{aligned}
\tag{2.4}
$$

However, a realistic description of the uncertainty generally requires a high number of scenarios. Therefore, the deterministic equivalent, problem (2.4), is often a large-scale problem, which necessitates the use of special-structure algorithms such as decomposition methods or Monte-Carlo simulations (see Birge and Louveaux [22] and Kall and Wallace [42] for an introduction to these techniques). Thus, problem (2.4) can be considerably harder to solve than problem (2.1), although both are linear. The difficulty in estimating probability distributions accurately also hinders the practical implementation of these techniques.

2.1.2 The Robust Approach

In contrast with the stochastic programming framework, robust optimization models random variables using uncertainty sets rather than probability distributions. The objective is then to minimize the worst-case cost in that set. Specifically, let $\mathcal{B}$ be the uncertainty set of the requirement vector having known mean $\overline{\mathbf{b}}$. The robust problem with recourse is formulated as:

$$\begin{aligned} \min \quad & \mathbf{c}^T\mathbf{x} + \max_{\mathbf{b}\in\mathcal{B}} Q(\mathbf{x},\mathbf{b}) \\ \text{s.t.} \quad & \mathbf{x} \in S. \end{aligned} \tag{2.5}$$

We assume relatively complete recourse (problem (2.2) is feasible for all $\mathbf{x} \in S$ and $\mathbf{b} \in \mathcal{B}$). Moreover, we assume for ease of presentation that $Q(\mathbf{x},\mathbf{b}) > -\infty$ for all $\mathbf{x} \in S$ and $\mathbf{b} \in \mathcal{B}$. By strong duality, we can write:

$$\begin{aligned} Q(\mathbf{x},\mathbf{b}) = \max_{\mathbf{p}} \quad & (\mathbf{b} - \mathbf{A}\mathbf{x})^T\mathbf{p} \\ \text{s.t.} \quad & \mathbf{B}^T\mathbf{p} \leq \mathbf{d}. \end{aligned} \tag{2.6}$$

Thus, problem (2.5) is equivalent to:

$$\min_{\mathbf{x}\in S}\left[\mathbf{c}^T\mathbf{x} + \max_{\mathbf{b}\in\mathcal{B},\mathbf{p}:\mathbf{B}^T\mathbf{p}\leq\mathbf{d}} (\mathbf{b} - \mathbf{A}\mathbf{x})^T\mathbf{p}\right]. \tag{2.7}$$

If $\mathcal{B} = \{\overline{\mathbf{b}}\}$, problem (2.5) is the "nominal" problem. As $\mathcal{B}$ expands around $\overline{\mathbf{b}}$, the decision-maker protects the system against more realizations of the random variables and the solution becomes more robust, but also more conservative. If the decision-maker does not take uncertainty into account, he might incur very large costs once the uncertainty has been revealed. On the other hand, if he includes every possible outcome in his model, he will protect the system against realizations that would indeed be detrimental to his profit, but are also very unlikely to happen. The question of choosing uncertainty sets that yield a good trade-off between performance and conservatism is central to robust optimization.

Following the approach developed by Bertsimas and Sim [14, 15] and Bertsimas and Thiele [18], we focus on polyhedral uncertainty sets and model the random variable b_i, $i = 1, \ldots, m$, as a parameter of known mean $\overline{b}_i$ and belonging to the interval $[\ \overline{b}_i - \widehat{b}_i,\ \overline{b}_i + \widehat{b}_i\ ]$. Equivalently:

$$b_i = \overline{b}_i + \widehat{b}_i\, z_i,\ \ |z_i| \leq 1,\ \ \forall i.$$

To avoid overprotecting the system, we impose the constraint:

$$\sum_{i=1}^{m} |z_i| \leq \Gamma,$$

which bounds the total scaled deviation of the parameters from their mean. Such a constraint was first proposed by Bertsimas and Sim [14] in the context of linear programming with uncertain coefficients. The parameter Γ, which we assume to be integer, is called the budget of uncertainty. $\Gamma = 0$ yields the nominal problem and, hence, does not incorporate uncertainty at all, while $\Gamma = m$ corresponds to interval-based uncertainty sets and leads to the most conservative case. In summary, we will consider the following uncertainty set:

$$\mathcal{B} = \left\{ \mathbf{b} : b_i = \overline{b}_i + \widehat{b}_i\ z_i,\ \ i = 1, \ldots, m,\ \ \mathbf{z} \in \mathcal{Z} \right\}, \tag{2.8}$$

with:

$$\mathcal{Z} = \left\{ \mathbf{z} : \sum_{i=1}^{m} |z_i| \leq \Gamma,\ \ |z_i| \leq 1,\ \ i = 1, \ldots, m \right\}. \tag{2.9}$$

In chapters II, III, and IV, we investigate how problem (2.5) can be solved efficiently (practically and theoretically) for the polyhedral set defined in equations (2.8)–(2.9), with an emphasis on the link with deterministic linear models and how the robust approach can help us gain insights into the impact of the uncertainty on the optimal solution.

2.2 Convex Adjustable Robust Optimization

As we propose to solve adjustable robust optimization problems using cutting-plane methods, we must have convexity of the adjustable robust problem. In this section, we prove that a general adjustable robust optimization problem, under mild assumptions, is a convex programming problem. Consider the following two-stage robust optimization problem:

$$\min_{\mathbf{x}\in S} \quad f(\mathbf{x}) + \max_{\mathbf{b}\in\mathcal{B}} Q(\mathbf{x},\mathbf{b}) \tag{2.10}$$

where

$$\begin{aligned} Q(\mathbf{x},\mathbf{b}) = \min_{\mathbf{y}\in Y} \quad & h(\mathbf{y}) \\ \text{s.t.} \quad & H(\mathbf{x},\mathbf{y},\mathbf{b}) \leq \mathbf{0}. \end{aligned} \tag{2.11}$$

The problem $\max_{\mathbf{b}\in\mathcal{B}} Q(\mathbf{x},\mathbf{b})$ is often referred to as the adversarial problem and the function $\mathcal{Q}(\mathbf{x})$, where $\mathcal{Q}(\mathbf{x}) = \max_{\mathbf{b}\in\mathcal{B}} Q(\mathbf{x},\mathbf{b})$, as the recourse function. The following list identifies the variables, parameters, and feasible decisions given in problems (2.10)–(2.11):

- $\mathbf{x}$, $\mathbf{y}$ are the first-stage and second-stage decisions, respectively,
- $\mathbf{b}$ is the vector of data,
- S, Y are the sets of all possible first-stage and second-stage decisions, respectively, and

- $\mathcal{B}$ is the uncertainty set.

We make a mild feasibility assumption that for any first-stage decision $\mathbf{x} \in S$ and any data instance $\mathbf{b} \in \mathcal{B}$, there exists $\mathbf{y} \in Y$ such that $H(\mathbf{x}, \mathbf{y}, \mathbf{b}) \leq 0$.

Proposition 2.12. *If the following conditions hold:*

A1 *S is a nonempty convex set,*

A2 *$f(\mathbf{x})$ is convex in $\mathbf{x}$,*

A3 *Y is a nonempty convex set,*

A4 *$h(\mathbf{y})$ is convex in $\mathbf{y}$,*

A5 *For all $i = 1, \ldots, p$, $H_i(\mathbf{x}, \mathbf{y}, \mathbf{b})$ is convex in $(\mathbf{x}, \mathbf{y})$, $\forall \mathbf{b} \in \mathcal{B}$,*

then problem (2.10) *is a convex optimization problem.*

Proof. First, note that for any $\mathbf{x}_1, \mathbf{x}_2 \in S$ and $\lambda \in [0, 1]$, we can take the convex combination $\lambda \mathbf{x}_1 + (1 - \lambda)\mathbf{x}_2$ due to **A1**. For problem (2.10) to be a convex programming problem we need the objective function to be convex in $\mathbf{x}$. We know that $f(\mathbf{x})$ is convex due to **A2**, so we have left to show that $\max_{\mathbf{b} \in \mathcal{B}} Q(\mathbf{x}, \mathbf{b})$ is convex in $\mathbf{x}$.

Let $\mathbf{b}$ be a fixed point such that $\mathbf{b} \in \mathcal{B}$. Given that we can find a feasible $\mathbf{y}$ for any fixed $\mathbf{b}$ and first-stage decision $\mathbf{x}$, we have the following: for any $\mathbf{x}_1, \mathbf{x}_2 \in S$, $Q(\mathbf{x}_i, \mathbf{b}) = h(\mathbf{y}_i)$ for $i = 1, 2$. Using the above we get the following.

$$
\begin{aligned}
Q(\lambda \mathbf{x}_1 + (1 - \lambda)\mathbf{x}_2, \mathbf{b}) &\leq h(\lambda \mathbf{y}_1 + (1 - \lambda)\mathbf{y}_2) \\
&\leq \lambda h(\mathbf{y}_1) + (1 - \lambda) h(\mathbf{y}_2) \\
&= \lambda Q(\mathbf{x}_1, \mathbf{b}) + (1 - \lambda) Q(\mathbf{x}_2, \mathbf{b})
\end{aligned}
$$

In the first inequality, we use conditions **A3** and **A5**, which give us convexity of the feasible region, i.e., we can write $\lambda \mathbf{y}_1 + (1 - \lambda)\mathbf{y}_2$ as a convex combination of $\mathbf{y}_1$ and

$\mathbf{y}_2$. The second inequality stems from condition **A4**, which gives us convexity in $\mathbf{y}$ of the inner minimization objective function $h(\mathbf{y})$. Conditions **A3-A5** are sufficient to ensure that $Q(\mathbf{x}, \mathbf{b})$ is convex in $\mathbf{x}$, $\forall \mathbf{b} \in \mathcal{B}$ and thus $\max_{\mathbf{b}\in\mathcal{B}} Q(\mathbf{x}, \mathbf{b})$ is convex in $\mathbf{x}$. $\square$

Let $f(\mathbf{x}) = \mathbf{c}^T\mathbf{x}$, $h(\mathbf{y}) = \mathbf{d}^T\mathbf{y}$, $Y = \{\mathbf{y} \geq \mathbf{0}\}$, and $H(\mathbf{x}, \mathbf{y}, \mathbf{b}) = \mathbf{A}\mathbf{x} - \mathbf{B}\mathbf{y} - \mathbf{b}$ (an equality constraint can be written as two inequality constraints and thus all equality constraints must be linear in $(\mathbf{x}, \mathbf{y})$ for any $\mathbf{b} \in \mathcal{B}$ to maintain convexity of $H(\mathbf{x}, \mathbf{y}, \mathbf{b})$ in $(\mathbf{x}, \mathbf{y})$ for any $\mathbf{b} \in \mathcal{B}$). Now that we have written problem (2.5) in the form of problem (2.10), we can apply Proposition 2.12 with the result that (2.5) is a convex programming problem and can be solved using cutting-plane methods.

2.3 Cutting-Plane Methods for Solving Convex Programming Problems

In this section, we describe three cutting-plane algorithms for solving problem (2.5): the first based on Kelley's algorithm, originally proposed in [43], secondly an analytic center cutting-plane method (ACCPM), and lastly a subgradient algorithm. We will describe in detail the algorithm based on Kelley's method, and only briefly describe the ACCPM and subgradient algorithm, because the modified Kelley's method performed exceedingly well and was far superior in solving two-stage linear programs with recourse than the ACCPM and the subgradient algorithm.

2.3.1 Kelley's Algorithm

Kelley's algorithm, as presented in [43], is designed to minimize a linear objective function over a compact convex feasible region that is complex (possibly described by an infinite number of constraints) or given only by a separation oracle, i.e., a subroutine that given a point in variable space, either correctly asserts that the point is feasible or returns the normal vector and intercept of some hyperplane that strictly separates the point from the feasible region. At each iteration, the algorithm maintains a polyhedral outer approximation of the feasible region. The objective function is minimized over the approximate feasible region, and if the arg min is infeasible, adds a linear inequality (a cut obtained from the separating hyperplane) to the approximate feasible region, thus improving the approximation. For problems with feasible regions described by a (possibly infinite) family of differentiable inequality constraints, cuts can be generated using gradients of the violated constraints.

Problem (2.5) has a simple feasible region S, but the objective function $\min \mathbf{c}^T\mathbf{x} + \max_{\mathbf{b}\in\mathcal{B}} Q(\mathbf{x}, \mathbf{b})$ is complex. Thus, in the implementation we are proposing, we focus on maintaining a piece-wise linear lower approximation of the objective function. The approximation is improved by adding cuts derived using subgradients of the objective function. The next section will provide a general outline of the version of Kelley's method we will be proposing, which will be further specialized for robust linear optimization with recourse.

Consider the following optimization problem:

$$\begin{aligned} \min_{\mathbf{x}} \quad & f(\mathbf{x}) \\ \text{s.t.} \quad & \mathbf{x} \in S \end{aligned} \tag{2.13}$$

where $f(\mathbf{x})$ is a convex function in $\mathbf{x}$, and S is a closed convex set. We assume that problem (2.13) has a finite, attainable optimal value. To implement all three algorithms to be discussed, we need to be able, given a value $\tilde{\mathbf{x}}$, to compute the value $f(\tilde{\mathbf{x}})$, as well as a subgradient $\mathbf{g}$ of $f(\mathbf{x})$ at $\tilde{\mathbf{x}}$ (denoted $\mathbf{g} \in \partial f(\tilde{\mathbf{x}})$), i.e., a vector $\mathbf{g}$ such that the following subgradient inequality is satisfied:

$$f(\mathbf{x}) \geq f(\tilde{\mathbf{x}}) + \mathbf{g}^T(\mathbf{x} - \tilde{\mathbf{x}}) \quad \forall \mathbf{x}. \tag{2.14}$$

In addition, we will maintain a lower bound L and an upper bound U on the optimal objective function value.

Algorithm 2.15. *(Kelley's Algorithm for problem* (2.13)*)*

Initialization: *Let $f_0(\mathbf{x})$ be an initial piece-wise linear lower approximation of $f(\mathbf{x})$. Set $L = -\infty$ and $U = \infty$; $t = 0$.*

Iteration t: *Given $f_t(\mathbf{x})$, L, and U,*

Step 1: *Solve $\min_{\mathbf{x}} f_t(\mathbf{x})$. Let $\mathbf{x}_t$ be an optimal solution and $L = f_t(\mathbf{x}_t)$.*

Step 2: *Compute $f(\mathbf{x}_t)$. Let $U = \min\{U, f(\mathbf{x}_t)\}$. If $U - L$ is sufficiently small, then stop and return $\mathbf{x}_t$ as the approximate solution to* (2.13).

Step 3: *Let $\mathbf{g}_t$ be a subgradient of $f(\mathbf{x})$ at $\mathbf{x}_t$. Define*

$$f_{t+1}(\mathbf{x}) = \max\{f_t(\mathbf{x}), f(\mathbf{x}_t) + \mathbf{g}_t^T(\mathbf{x} - \mathbf{x}_t)\}.$$

Step 4: *Set $t \leftarrow t + 1$.*

Note that the cut added to the piecewise linear lower approximation at iteration t is a supporting hyperplane to the epigraph of function $f(\mathbf{x})$ and it separates the point $(\mathbf{x}_t, f_t(\mathbf{x}_t))$ from the epigraph.

2.3.2 Analytic Center Cutting Plane Method

The analytic center cutting-plane method (ACCPM) is an example of an interior-point cutting-plane method, which has been proven effective in terms of both theoretical complexity [2, 39, 46, 48] and practical performance [4, 45, 46], and other references therein, on a variety of problems.

At the beginning of a typical iteration of the ACCPM, we have available a set P known to contain the feasible region and an upper bound U on the optimal objective function value. The algorithm proceeds by finding the analytic center $\mathbf{x}$ of the set

$P \cap \{\mathbf{x} \mid f(\mathbf{x}) \leq U\}$ and calling the separation oracle for $\mathbf{x}$. If $\mathbf{x}$ is feasible, then the upper bound is reset to be $U := f(\mathbf{x})$ (since the analytic center cannot lie on the boundary of the region, $f(\mathbf{x}) \leq U$). Otherwise, the valid inequality obtained from the separating hyperplane provided by the oracle is added to the description of the polyhedron P. The iteration is then repeated until appropriate termination criteria are satisfied. Note that to implement the ACCPM, the set $P \cap \{\mathbf{x} \mid f(\mathbf{x}) \leq U\}$ must be such that its analytic center can be computed efficiently.

Define S in problem (2.13) as $S = \{\mathbf{x} \mid f_i(\mathbf{x}) \leq 0,\ i = 1, \ldots, \tilde{m}\}$, where $f_1, \ldots, f_{\tilde{m}}$ are convex functions in $\mathbf{x}$ leading us to the following general convex programming problem.

$$\begin{aligned} \min_{\mathbf{x}} \quad & f(\mathbf{x}) \\ \text{s.t.} \quad & f_i(\mathbf{x}) \leq 0 \quad i = 1, \ldots, \tilde{m} \end{aligned} \tag{2.16}$$

Algorithm 2.17. *(ACCPM for problem* (2.16)*)*

Initialization: *Let P_0 be some polyhedron that is known to contain S:*

$$S \subseteq P_0 = \{\mathbf{x} \mid l_i(\mathbf{x}) \leq 0,\ i = 1, \ldots, \tilde{L}\},$$

where $l_i(\mathbf{x})$ are linear functions in $\mathbf{x}$. Let $L = -\infty$ and $U = \infty$.

Iteration t: *Given P_t, L, and U,*

Step 1: *Compute $\mathbf{x}_t$ as the analytic center of P_t,*

$$\mathbf{x}_t = arg \min_{\mathbf{x}} \Big\{ -\sum_{i=1}^{\tilde{L}} \log(-l_i(\mathbf{x})) - \sum_{i=1}^{t-1} \log(-\tilde{f}_i(\mathbf{x})) \Big\},$$

where $\tilde{f}_i(\mathbf{x})$ are previously added feasibility and/or optimality cuts. The lower bound L must be computed by a separate subroutine, implemented here. Let L be the output of this subroutine evaluated at the analytic center $\mathbf{x}_t$.

Step 2: *Check feasibility of* $\mathbf{x}_t$. *If* $\mathbf{x}_t$ *violates some constraint* $f_i(\mathbf{x})$, *i.e.* $f_i(\mathbf{x}_t) > 0$ *for some* i, *then add the following feasibility cut:*

$$P_{t+1} = P_t \cap \{\mathbf{x} \mid \tilde{f}_t = f_i(\mathbf{x}_t) + \mathbf{g}_t^T(\mathbf{x} - \mathbf{x}_t) \leq 0\},$$

where $\mathbf{g}_t$ *is a subgradient of* $f_i(\mathbf{x})$ *evaluated at* $\mathbf{x}_t$. *Skip to Step 5. Else go to Step 3.*

Step 3: *Compute* $f(\mathbf{x}_t)$. *Let* $U = \min\{U, f(\mathbf{x}_t)\}$. *If* $U - L$ *is sufficiently small, then stop and return* $\mathbf{x}_t$ *as the approximate solution.*

Step 4: *If* $\mathbf{x}_t$ *is feasible, then add the following optimality cut:*

$$P_{t+1} = P_t \cap \{\mathbf{x} \mid \tilde{f}_t = \mathbf{g}_t^T(\mathbf{x} - \mathbf{x}_t) \leq 0\},$$

where $\mathbf{g}_t$ *is a subgradient of* $f(\mathbf{x})$ *evaluated at* $\mathbf{x}_t$.

Step 5: *Set* $t \leftarrow t + 1$.

To prove that the above feasibility cut only cuts off points that are infeasible, and not points that are feasible, we want to show that given a feasible $\mathbf{x}$ and an infeasible $\mathbf{x}_t$, $f_i(\mathbf{x}_t) + \mathbf{g}_t^T(\mathbf{x} - \mathbf{x}_t) \leq 0$. We know that for any feasible $\mathbf{x}$, $f_i(\mathbf{x}) \leq 0$. We also have that $f_i(\mathbf{x}) \geq f_i(\mathbf{x}_t) + \mathbf{g}_t^T(\mathbf{x} - \mathbf{x}_t)$ by the gradient inequality since $f_i(\mathbf{x})$ is a convex function. This gives us the following:

$$0 \geq f_i(\mathbf{x}) \geq f_i(\mathbf{x}_t) + \mathbf{g}_t^T(\mathbf{x} - \mathbf{x}_t)$$

and thus $0 \geq f_i(\mathbf{x}_t) + \mathbf{g}_t^T(\mathbf{x} - \mathbf{x}_t)$.

To prove that an optimality cut only cuts off points that have worse objective function values than $\mathbf{x}_t$ and not points that have equal or better objective function values than $\mathbf{x}_t$, we want to show that given a feasible $\mathbf{x}$, $\mathbf{g}_t^T(\mathbf{x} - \mathbf{x}_t) \leq 0$. Again, since $f(\mathbf{x})$ is a convex function, the gradient inequality will hold: $f(\mathbf{x}) \geq f(\mathbf{x}_t) + \mathbf{g}_t^T(\mathbf{x} - \mathbf{x}_t)$.

We can rewrite this as $f(\mathbf{x}) - f(\mathbf{x}_t) \geq \mathbf{g}_t^T(\mathbf{x} - \mathbf{x}_t)$. Additionally, since we are trying to cut off points with worse objective function values than $\mathbf{x}_t$, for all feasible $\mathbf{x}$, the following will hold: $f(\mathbf{x}) - f(\mathbf{x}_t) \leq 0$. Note that the problem is a minimization problem so a better objective function value in this case means a smaller value. This gives us the following:

$$\mathbf{g}_t^T(\mathbf{x} - \mathbf{x}_t) \leq f(\mathbf{x}) - f(\mathbf{x}_t) \leq 0$$

and thus $\mathbf{g}_t^T(\mathbf{x} - \mathbf{x}_t) \leq 0$.

2.3.3 Subgradient Algorithm

The subgradient algorithm is a steepest-descent-like algorithm, which can be used to solve convex optimization problems with non-differentiable objective functions. When the objective function is differentiable, then the subgradient algorithm for unconstrained optimization will use the same direction as the steepest-descent method. (See Bazaraa et al. [5] Section 8.9 for a discussion of the subgradient algorithm presented here along with some of the implementation difficulties one must consider.)

Consider the standard convex program, problem (2.13). At the beginning of a typical iteration of the subgradient algorithm, we have a point $\mathbf{x}_t \in S$. The algorithm proceeds to find another point $\mathbf{x}_{t+1} \in S$ such that $f(\mathbf{x}_{t+1}) < f(\mathbf{x}_t)$ and does so by computing a step size μ_t, finding a subgradient $\mathbf{g}_t$ of $f(\mathbf{x})$ evaluated at $\mathbf{x}_t$, and starting from $\mathbf{x}_t$, taking a step of length μ_t in the negative direction of $\mathbf{g}_t$. Given that problem (2.13) is a constrained optimization problem, we must make sure to maintain feasibility in each iteration of the subgradient algorithm. If once you arrive at your new point x_{t+1} and $\mathbf{x}_{t+1} \notin S$, then you can either project x_{t+1} onto S or backtrack in the direction of the subgradient (shrinking the step size μ until you reach the feasible region). In our implementation, we used backtracking to maintain feasibility. The iteration is repeated until appropriate termination criteria are satisfied.

Algorithm 2.18. *(Subgradient algorithm for problem* (2.13)*)*

Initialization: *Let* $\mathbf{x}_0$ *be the starting point and* μ_0 *be the initial starting step size.*

Iteration t**:** *Given* $\mathbf{x}_t$ *and* μ_t*,*

> **Step 1:** *Find* $\mathbf{g}_t$ *evaluated at* $\mathbf{x}_t$*. If* $\mathbf{g}_t = \mathbf{0}$*, then stop.*
>
> **Step 2:** $\bar{\mathbf{x}}_{t+1} = \mathbf{x}_t - \mu_t \mathbf{g}_t$.
>
> **Step 3:** *If* $\bar{\mathbf{x}}_{t+1} \notin S$*, backtrack (shrink* μ_t*) until* $\mathbf{x}_t - \mu_t \mathbf{g}_t \in S$*. If* $\bar{\mathbf{x}}_{t+1} \in S$*, then* $\mathbf{x}_{t+1} = \bar{\mathbf{x}}_{t+1}$.
>
> **Step 4:** *Compute* $f(\mathbf{x}_{t+1})$*. Let* $U = \min\{U, f(\mathbf{x}_{t+1})\}$.
>
> **Step 5:** *Set* $t \leftarrow t + 1$.

While the subgradient algorithm provides an overall framework, what remains to be specified is the step size in each iteration. The step size in our implementation can be found in Wolsey [66] Theorem 10.4(c), which has guaranteed convergence, and involves a difference of bounds (upper bound minus lower bound) over the squared norm of the subgradient evaluated at the current iterate. The lower bound is obtained by solving the approximate problem found in Step 1 of Algorithm 2.15 and using the cuts generated by the subgradient algorithm.

2.4 Discussion of Difficulties in Applying Cutting-Plane Methods to Adjustable Robust Problems

To remind the reader, the problem we wish to solve is problem (2.7), presented again below:

$$\min_{\mathbf{x} \in S} \left[\mathbf{c}^T \mathbf{x} + \max_{\mathbf{b} \in \mathcal{B}, \mathbf{p}: \mathbf{B}^T \mathbf{p} \leq \mathbf{d}} (\mathbf{b} - \mathbf{A}\mathbf{x})^T \mathbf{p} \right].$$

To compute $\mathbf{c}^T\mathbf{x}+\max_{\mathbf{b}\in\mathcal{B}} Q(\mathbf{x},\mathbf{b})$ (the true objective function value at a given point), given first-stage decision $\mathbf{x}_t$, we need to be able to solve the adversarial problem:

$$\begin{aligned}\max_{\mathbf{b}\in\mathcal{B}} Q(\mathbf{x}_t,\mathbf{b}) &= \max_{\mathbf{b}\in\mathcal{B}}\min_{\mathbf{y}\in Y}\{\mathbf{d}^T\mathbf{y} \mid \mathbf{B}\mathbf{y}=\mathbf{b}-\mathbf{A}\mathbf{x}_t\}\\ &= \max_{\mathbf{b},\mathbf{p}}\{\mathbf{b}^T\mathbf{p}-\mathbf{x}_t^T\mathbf{A}^T\mathbf{p} \mid \mathbf{B}^T\mathbf{p}\le\mathbf{d}, \mathbf{b}\in\mathcal{B}\}\end{aligned}$$

Note that the adversarial problem, when viewed through the primal representation, is a max-min formulation and thus a saddle-point problem, while the adversarial problem, when viewed through the dual representation, is a non-convex quadratic optimization problem. Additionally, since $Q(\mathbf{x},\mathbf{b})$ is convex in $\mathbf{b}$ for any $\mathbf{x}$, it requires maximization of a convex function. In general, the adversarial problem is a difficult problem to solve; however, there are some special cases for which we can solve the adversarial problem fairly easily, which will be discussed in the following subsection. How we solve the adversarial problem for program (2.7) will be discussed in detail for simple and general recourse in Chapters III and IV.

2.4.1 When the Adversarial Problem is Easily Solved

If $\mathcal{B}$ is a finite set of points $\{\mathbf{b}_1,\ldots,\mathbf{b}_\Omega\}$ or a convex set expressed as the convex hull of a known list of extreme points $\{\mathbf{b}_1,\ldots,\mathbf{b}_\Omega\}$, then the maximum of $Q(\mathbf{x}_t,\mathbf{b})$ over $\mathcal{B}$ is attained at one of the points $\mathbf{b}_\omega$, $\omega=1,\ldots,\Omega$, which results in the following:

$$\max_{\mathbf{b}\in\mathcal{B}} Q(\mathbf{x}_t,\mathbf{b}) = \max_{\omega=1,\ldots,\Omega} Q(\mathbf{x}_t,\mathbf{b}_\omega) = \max_{\omega=1,\ldots,\Omega}\max_{\mathbf{p}}\{(\mathbf{b}_\omega-\mathbf{A}\mathbf{x}_t)^T\mathbf{p} \mid \mathbf{B}^T\mathbf{p}\le\mathbf{d}\}.$$

If Ω is of reasonable size and the polyhedron $\{\mathbf{p} \mid \mathbf{B}^T\mathbf{p}\le\mathbf{d}\}$ is easy to optimize over, the inner maximization in the last expression can be done easily for each $\mathbf{b}_\omega$, and then the maximum over ω can be taken to obtain the solution to the adversarial problem.

Another instance when the adversarial problem can be solved easily is when the polyhedral set $\{\mathbf{p} \mid \mathbf{B}^T\mathbf{p}\le\mathbf{d}\}$ is bounded and its extreme points are a known list given as $\{\mathbf{p}_1, \ldots, \mathbf{p}_\Delta\}$ with Δ being of reasonable size. Then we can rewrite the adversarial problem as follows:

$$\begin{aligned}\max_{\mathbf{b}\in\mathcal{B}} \max_{\delta=1,\ldots,\Delta} (\mathbf{b} - \mathbf{A}\mathbf{x}_t^T)\mathbf{p}_\delta &= \max_{\delta=1,\ldots,\Delta} \max_{\mathbf{b}\in\mathcal{B}} (\mathbf{b} - \mathbf{A}\mathbf{x}_t)^T \mathbf{p}_\delta \\ &= \max_{\delta=1,\ldots,\Delta} \left\{ \left(\max_{\mathbf{b}\in\mathcal{B}} \mathbf{b}^T \mathbf{p}_\delta \right) - \mathbf{x}_t^T \mathbf{A}^T \mathbf{p}_\delta \right\}.\end{aligned}$$

If the situation is such that a linear function can be easily optimized over $\mathcal{B}$, then the inner maximization in the last expression can be done easily for each $\mathbf{p}_\delta$ and then the maximum over δ can be taken to obtain the solution to the adversarial problem. If both sets $\mathcal{B}$ and $\{\mathbf{p} \mid \mathbf{B}^T\mathbf{p} \leq \mathbf{d}\}$ can be described by a list of their extreme points then we can solve the adversarial problem as follows:

$$\max_{\mathbf{b}\in\mathcal{B}} Q(\mathbf{x}_t, \mathbf{b}) = \max_{\omega=1,\ldots,\Omega} \max_{\delta=1,\ldots,\Delta} (\mathbf{b}_\omega - \mathbf{A}\mathbf{x}_t)^T \mathbf{p}_\delta.$$

2.4.2 Form of Subgradient for Adding Cuts

We give the form for the subgradient of problem (2.5) and then discuss how this subgradient fits into Algorithms 2.15, 2.17, and 2.18. Let $f(\mathbf{x}) = \mathbf{c}^T\mathbf{x} + \max_{\mathbf{b}\in\mathcal{B}} Q(\mathbf{x}, \mathbf{b})$.

Lemma 2.19. *Let* $\bar{\mathbf{b}} \in \arg\max_{\mathbf{b}\in\mathcal{B}} Q(\bar{\mathbf{x}}, \mathbf{b})$. *Furthermore, let* $\bar{\mathbf{p}}$ *be an optimal solution of* (2.6) *with* $(\mathbf{x}, \mathbf{b}) = (\bar{\mathbf{x}}, \bar{\mathbf{b}})$. *Then* $(\mathbf{c}^T - \mathbf{A}^T\bar{\mathbf{p}}) \in \partial f(\bar{\mathbf{x}})$.

Proof. For an arbitrary $\mathbf{x}$,

$$\begin{aligned} f(\mathbf{x}) &= \mathbf{c}^T\mathbf{x} + \mathcal{Q}(\mathbf{x}) \geq \mathbf{c}^T\mathbf{x} + Q(\mathbf{x}, \bar{\mathbf{b}}) \\ &\geq \mathbf{c}^T\mathbf{x} + (\bar{\mathbf{b}} - \mathbf{A}\mathbf{x})^T\bar{\mathbf{p}} \\ &= \mathbf{c}^T\bar{\mathbf{x}} + Q(\bar{\mathbf{x}}, \bar{\mathbf{b}}) + (\mathbf{c} - \mathbf{A}^T\bar{\mathbf{p}})^T(\mathbf{x} - \bar{\mathbf{x}}) \\ &= f(\bar{\mathbf{x}}) + (\mathbf{c} - \mathbf{A}^T\bar{\mathbf{p}})^T(\mathbf{x} - \bar{\mathbf{x}}), \end{aligned}$$

proving the claim. $\square$

Using the result of Lemma 2.19, we can now discuss how the subgradient fits into each algorithm. Step 3 in Algorithm 2.15 will take the following form:

$$f_{t+1}(\mathbf{x}) = \max\{f_t(\mathbf{x}), f(\mathbf{x}_t) + (\mathbf{c} - \mathbf{A}^T\mathbf{p})^T(\mathbf{x} - \mathbf{x}_t)\}.$$

Step 2 in Algorithm 2.17 is dependent upon the form of the inequalities describing S and thus cannot be characterized any further. However, Step 4 in Algorithm 2.17 will take the following form:

$$P_{t+1} = P_t \cap \{\mathbf{x} \mid \tilde{f}_t = (\mathbf{c} - \mathbf{A}^T\mathbf{p})^T(\mathbf{x} - \mathbf{x}_t) \leq 0\}.$$

Lastly, Step 1 of Algorithm 2.18 will find a subgradient of the form $(\mathbf{c} - \mathbf{A}^T\mathbf{p})$.

2.5 Kelley's Algorithm for Robust Linear Programming with Recourse

As mentioned earlier, we will focus on Kelley's algorithm as the solution method for solving robust linear programming problems with recourse and will now specify algorithm 2.15 (Kelley's algorithm) for problem (2.7).

Algorithm 2.20. *(Kelley's Algorithm for Robust Linear Program with Recourse)*

Initialization: *Let* $\mathcal{Q}_0(\mathbf{x})$ *be the initial piecewise linear lower approximation of* $\mathcal{Q}(\mathbf{x})$*.*

Set $L = -\infty$ *and* $U = \infty$*;* $t = 0$*.*

Iteration t: *Given* L*,* U*, and* $\mathcal{Q}_t(\mathbf{x})$*,*

Step 1: *Solve* $\min_{\mathbf{x} \in S} \mathbf{c}^T\mathbf{x} + \mathcal{Q}_t(\mathbf{x})$*:*

$$\begin{aligned} \min_{\mathbf{x},\alpha} \quad & \mathbf{c}^T\mathbf{x} + \alpha \\ s.t. \quad & \alpha + \mathbf{p}_l^T\mathbf{A}\mathbf{x} \geq \mathbf{b}_l^T\mathbf{p}_l, \quad l = 1, \ldots, t-1 \\ & \alpha \geq \mathcal{Q}_0(\mathbf{x}) \\ & \mathbf{x} \in S. \end{aligned} \tag{2.21}$$

Let $(\mathbf{x}_t, \alpha_t)$ *be an optimal solution and let* $L = \mathbf{c}^T\mathbf{x}_t + \mathcal{Q}_t(\mathbf{x}_t)$*.*

Step 2: *Compute* $\mathcal{Q}(\mathbf{x}_t)$*, let* $\mathbf{b}_t$ *and* $\mathbf{p}_t$ *be the corresponding worst-case demand and dual recourse vector, respectively. Let* $U = \min\{U, \mathbf{c}^T\mathbf{x}_t + \mathcal{Q}(\mathbf{x}_t)\}$*. If* $U - L$ *is sufficiently small, stop and return* $\mathbf{x}_t$ *as an approximate solution.*

Step 3: *Define*

$$\mathcal{Q}_{t+1}(\mathbf{x}) = \max\{\mathcal{Q}(\mathbf{x}_t), \mathbf{p}_t^T(\mathbf{b}_t - \mathbf{A}\mathbf{x})\}.$$

Step 4: *Set* $t \leftarrow t + 1$*.*

Observe that, for a given $\mathbf{x}$, the function $\max_{\mathbf{b}\in\mathcal{B}}(\mathbf{b} - \mathbf{A}\mathbf{x})^T\mathbf{p}$ is convex in $\mathbf{p}$, and therefore problem (2.5) can be rewritten as the following master problem:

$$\begin{aligned} \min_{\mathbf{x},\alpha} \quad & \mathbf{c}^T\mathbf{x} + \alpha \\ \text{s.t.} \quad & \alpha \geq \max_{\mathbf{b}\in\mathcal{B}}(\mathbf{b} - \mathbf{A}\mathbf{x})^T\mathbf{p}_k, \quad k = 1, \ldots, K \\ & \mathbf{x} \in S, \end{aligned} \tag{2.22}$$

where $p_k,\ k = 1, \ldots, K$ are the extreme points of $\{\mathbf{p} \mid \mathbf{B}^T\mathbf{p} \leq \mathbf{d}\}$. Algorithm 2.20 can be seen as a variant of delayed constraint generation for problem (2.22), with relaxed master problem (2.21), and convergence of the algorithm follows from this observation. Bertsimas and Tsitsiklis provide an introduction to these techniques in [19]. (The reader is also referred to Birge and Louveaux [22] and Kall and Wallace [42] for an extensive treatment of these methods in the context of stochastic optimization.)

Application of the delayed constraint generation technique to the stochastic programming problem (2.4) is referred to as Benders' decomposition [13]. The corresponding master problem can be written as:

$$\begin{aligned} \min \quad & \mathbf{c}^T\mathbf{x} + \sum_{\omega=1}^{\Omega} \pi_\omega Z_\omega \\ \text{s.t.} \quad & Z_\omega \geq \mathbf{p}_k^T(\mathbf{b}_\omega - \mathbf{A}\mathbf{x}) \quad \forall k, \omega \\ & \mathbf{x} \geq \mathbf{0}, \end{aligned}$$

where $\omega = 1, \ldots, \Omega$ are the scenarios. Here at each iteration, a relaxed master problem is solved to obtain a first-stage solution $\tilde{x}$ and the corresponding value of the recourse function $\tilde{Z}_\omega$ when scenario ω is realized. To check if this solution is optimal for the full master problem or to apply a cut to the expected recourse function $\sum_{\omega=1}^{\Omega} \pi_\omega Q(\mathbf{x}, \mathbf{b}_\omega)$, one needs to solve the recourse problem (2.4) for each scenario $\omega = 1, \ldots, \Omega$. While these problems are similar to each other and each can be solved efficiently by applying, for instance, the dual simplex method, the large number of subproblems is a drawback in accurately solving the stochastic programming counterpart of problem (2.1) in many real-life settings. In contrast, Algorithm 2.20, which applies a similar technique to the adjustable robust counterpart of problem (2.1), involves solving only one subproblem per iteration. This plays a key role in the tractability of the robust approach in all settings where the relevant subproblem can be identified efficiently, which will be discussed in Chapters III and IV.

CHAPTER III

Analysis of Robust Linear Optimization Problems with Simple Recourse and Computational Experiments

In this chapter, we present a variety of experimental results when using Kelley's Method to solve a newsvendor problem with simple recourse including an analysis of problem solutions and performance results of Kelley's Method. [1]

3.1 Analysis of Robust Linear Programs with Simple Recourse

In linear programs with simple recourse, the decision-maker is able to address excess or shortage for each of the requirements independently. For instance, he might pay a unit shortage penalty s_i for falling short of the random target b_i or a unit holding cost h_i for exceeding the random target b_i, for each i. We describe an application of this setting to multi-item newsvendor problems in Section 3.2.

The deterministic model can be formulated as:

$$\begin{aligned} \min \quad & \mathbf{c}^T\mathbf{x} + \mathbf{s}^T\mathbf{y}^- + \mathbf{h}^T\mathbf{y}^+ \\ \text{s.t.} \quad & \mathbf{A}\mathbf{x} + \mathbf{y}^- - \mathbf{y}^+ = \mathbf{b}, \\ & \mathbf{x} \in S,\ \mathbf{y}^-,\ \mathbf{y}^+ \geq \mathbf{0}, \end{aligned}$$

[1] Some of the analysis and experimental results presented in this chapter can also be found in Thiele et al. [63].

and the recourse function defined in equation 2.2 becomes:

$$
\begin{aligned}
Q(\mathbf{x},\mathbf{b}) = \quad \min \quad & \mathbf{s}^T\mathbf{y}^- + \mathbf{h}^T\mathbf{y}^+ \\
\text{s.t.} \quad & \mathbf{y}^- - \mathbf{y}^+ = \mathbf{b} - \mathbf{A}\mathbf{x}, \\
& \mathbf{y}^-,\ \mathbf{y}^+ \geq \mathbf{0},
\end{aligned} \tag{3.1}
$$

We will require that $\mathbf{s}+\mathbf{h} \geq \mathbf{0}$ to ensure finiteness of the recourse function. It is straightforward to see that $Q(\mathbf{x},\mathbf{b})$ is available in closed form:

$$
Q(\mathbf{x},\mathbf{b}) = \sum_{i=1}^{m} \big[s_i \cdot \max\{0, b_i - (\mathbf{A}\mathbf{x})_i\} + h_i \cdot \max\{0, (\mathbf{A}\mathbf{x})_i - b_i\} \big].
$$

However, we will focus on problem 3.1 to build a tractable robust model. We obtain an equivalent characterization of the recourse function by invoking strong duality:

$$
\begin{aligned}
Q(\mathbf{x},\mathbf{b}) = \quad \max \quad & (\mathbf{b} - \mathbf{A}\mathbf{x})^T\mathbf{p} \\
\text{s.t.} \quad & -\mathbf{h} \leq \mathbf{p} \leq \mathbf{s}.
\end{aligned} \tag{3.2}
$$

Therefore, in this section we will be developing efficient ways to solve:

$$
\min_{\mathbf{x}\in S} \quad \left[\mathbf{c}^T\mathbf{x} + \max_{\mathbf{b}\in\mathcal{B},\ -\mathbf{h}\leq\mathbf{p}\leq\mathbf{s}} (\mathbf{b} - \mathbf{A}\mathbf{x})^T\mathbf{p} \right], \tag{3.3}
$$

where $\mathcal{B}$ has been defined in equations 2.8–2.9.

3.1.1 Computing $\mathcal{Q}(\mathbf{x})$ for Robust Linear Programs with Simple Recourse

The following theorem provides a simple method for computing $\mathcal{Q}(\mathbf{x})$ in problems with simple recourse. In the proof, we refer to the set

$$
\mathcal{Z}' = \Big\{ \mathbf{z}' \mid \sum_{i=1}^{m} z'_i \leq \Gamma, 0 \leq z'_i \leq 1, i = 1,\ldots,m \Big\}. \tag{3.4}
$$

Theorem 3.5 (Calculating $\mathcal{Q}(\mathbf{x})$). *Given* $\mathbf{x}$, *define for* $i = 1,\ldots,m$,

$$
\begin{aligned}
\Delta_i = \max \Big\{ & (\bar{b}_i + \widehat{b}_i - (\mathbf{A}\mathbf{x})_i)\ s_i, ((\mathbf{A}\mathbf{x})_i - \bar{b}_i + \widehat{b}_i)\ h_i \Big\} \\
& - \max \big\{ (\bar{b}_i - (\mathbf{A}\mathbf{x})_i)\ s_i, ((\mathbf{A}\mathbf{x})_i - \bar{b}_i)\ h_i \big\}.
\end{aligned} \tag{3.6}
$$

Let $\mathcal{I}$ be the set of indices corresponding to the Γ-largest Δ_i. Then $\mathcal{Q}(\mathbf{x}) = \max_{\mathbf{b}\in\mathcal{B}} Q(\mathbf{x}, \mathbf{b})$ verifies:

$$\begin{aligned}\mathcal{Q}(\mathbf{x}) = \sum_{i\in\mathcal{I}} \max\Big\{(\bar{b}_i + \widehat{b}_i - (\mathbf{Ax})_i)\ s_i, ((\mathbf{Ax})_i - \bar{b}_i + \widehat{b}_i)\ h_i\Big\} \\ + \sum_{i\notin\mathcal{I}} \max\left\{(\bar{b}_i - (\mathbf{Ax})_i)\ s_i, ((\mathbf{Ax})_i - \bar{b}_i)\ h_i\right\}.\end{aligned} \tag{3.7}$$

Proof. We note that for any first-stage decision vector $\mathbf{x}$:

$$\mathcal{Q}(\mathbf{x}) = \max_{\mathbf{b}\in\mathcal{B}} \max_{-\mathbf{h}\leq p\leq \mathbf{s}} (\mathbf{b} - \mathbf{Ax})^T\mathbf{p}, \tag{3.8}$$

$$= \max_{\mathbf{b}\in\mathcal{B}} \sum_{i=1}^{m} \max\left\{(b_i - (\mathbf{Ax})_i)\ s_i, ((\mathbf{Ax})_i - b_i)\ h_i\right\}, \tag{3.9}$$

$$= \sum_{i=1}^{m} \max\left\{(\bar{b}_i - (\mathbf{Ax})_i)\ s_i, ((\mathbf{Ax})_i - \bar{b}_i)\ h_i\right\} + \max_{\mathbf{z}'\in\mathcal{Z}'} \sum_{i=1}^{m} \Delta_i\ z'_i, \tag{3.10}$$

where $\mathcal{Z}'$ is defined in equation (3.4). The last equality is obtained by observing that the expression in (3.9) is convex in $\mathbf{b}$, hence the worst-case value of $\mathbf{b}$ that attains the maximum can be found at an extreme point of $\mathcal{B}$. The extreme points of $\mathcal{B}$ can be enumerated by letting Γ components of $\mathbf{b}$ deviate up or down (to their highest or lowest values), while keeping the remaining components at their nominal values. Whether the worst case is reached when b_i deviates up or down (to its highest or lowest value) is captured by the value of Δ_i. It then follows that $\max_{\mathbf{z}'\in\mathcal{Z}'} \sum_{i=1}^{m} \Delta_i\ z'_i$ is equal to $\sum_{i\in\mathcal{I}} \Delta_i$. □

Corollary 3.11. *Given $\mathbf{x}$, the corresponding worst-case demand $\mathbf{b}$ can be computed as follows: for $i = 1, \ldots, m$*

$$b_i = \begin{cases} \bar{b}_i + \hat{b}_i & \text{if } i \in \mathcal{I} \text{ and } (\bar{b}_i + \hat{b}_i - (\mathbf{Ax})_i)\ s_i \geq ((\mathbf{Ax})_i - \bar{b}_i + \hat{b}_i)\ h_i \\ \bar{b}_i - \hat{b}_i & \text{if } i \in \mathcal{I} \text{ and } (\bar{b}_i + \hat{b}_i - (\mathbf{Ax})_i)\ s_i < ((\mathbf{Ax})_i - \bar{b}_i + \hat{b}_i)\ h_i \\ \bar{b}_i & \text{if } i \notin \mathcal{I}. \end{cases} \tag{3.12}$$

The corresponding dual recourse vector $\mathbf{p}$ *can be determined as follows: for* $i = 1, \ldots, m$

$$p_i = \begin{cases} s_i & \text{if } i \in \mathcal{I} \text{ and } (\bar{b}_i + \hat{b}_i - (\mathbf{Ax})_i)\ s_i \geq ((\mathbf{Ax})_i - \bar{b}_i + \hat{b}_i)\ h_i \\ -h_i & \text{if } i \in \mathcal{I} \text{ and } (\bar{b}_i + \hat{b}_i - (\mathbf{Ax})_i)\ s_i < ((\mathbf{Ax})_i - \bar{b}_i + \hat{b}_i)\ h_i \\ s_i & \text{if } i \notin \mathcal{I} \text{ and } (\bar{b}_i - (\mathbf{Ax})_i)\ s_i \geq (\mathbf{Ax})_i - \bar{b}_i)\ h_i \\ -h_i & \text{if } i \notin \mathcal{I} \text{ and } (\bar{b}_i - (\mathbf{Ax})_i)\ s_i < ((\mathbf{Ax})_i - \bar{b}_i)\ h_i. \end{cases} \tag{3.13}$$

The subgradient of $\mathcal{Q}(\mathbf{x})$ can now be computed as in Lemma 2.19, allowing the implementation of Kelley's method, Algorithm 2.20.

3.2 Computational Results: Newsvendor Problem

In this section, we test the robust methodology on a multi-item newsvendor problem. The decision-maker orders perishable items subject to a capacity constraint, faces uncertain demand, and incurs surplus and shortage costs for each item at the end of the time period. His goal is to minimize total cost. We use the following notation:

n : the number of items,

c_i : the unit ordering cost of item i,

h_i : the unit holding cost of item i,

s_i : the unit shortage cost of item i,

b_i : the demand for item i,

A : the purchasing budget.

The deterministic problem can be formulated as:

$$\begin{aligned} \min \quad & \mathbf{c}^T\mathbf{x} + \sum_{i=1}^{n} \max\left\{s_i\ (b_i - x_i), h_i\ (x_i - b_i)\right\} \\ \text{s.t.} \quad & \mathbf{c}^T\mathbf{x} \le A, \\ & \mathbf{x} \ge \mathbf{0} \end{aligned}$$

or equivalently as:

$$\begin{aligned} \min \quad & \mathbf{c}^T\mathbf{x} + \mathbf{s}^T\mathbf{y}^- + \mathbf{h}^T\mathbf{y}^+ \\ \text{s.t.} \quad & \mathbf{x} + \mathbf{y}^- - \mathbf{y}^+ = \mathbf{b}, \\ & \mathbf{c}^T\mathbf{x} \le A, \\ & \mathbf{x} \ge \mathbf{0}. \end{aligned} \tag{3.14}$$

Problem (3.14) is an example of a linear programming problem with simple recourse and therefore can be analyzed using the techniques described in Sections 2.3.1. We consider a case with $n = 50$ items and budget $A = 5000$, with ordering cost $c_i = 1$, nominal demand $\bar{b}_i = 8 + 2\,i$, and maximum deviation of the demand from its nominal value $\widehat{b}_i = 0.5 \cdot \bar{b}_i$, for each $i = 1, \ldots, 50$. We consider two different structures for the surplus and shortage penalties, resulting in two instances of problem (3.14). In the first instance, items with larger nominal demand (and thus wider demand variability by the above definitions of $\overline{\mathbf{b}}$ and $\widehat{\mathbf{b}}$) have larger surplus and shortage penalties than items with small nominal demand. In the second instance, surplus and shortage penalties follow the opposite pattern. In particular, the penalties for item i are shown in the table below:

	Shortage s_i	Holding h_i
Instance 1	$2i$	i
Instance 2	$2(n+1-i)$	$(n+1-i)$

Table 3.1: Surplus and shortage penalties for item i for two instances of the newsvendor problem.

We applied Algorithm 2.15 to problem (3.14) using AMPL/CPLEX v.10.0. Since the recourse value in this problem is always nonnegative, we set $\mathcal{Q}_0(\mathbf{x}) \equiv 0$ in the initialization step. Step 2 of the algorithm was carried out as discussed in Theorem 3.5 and Corollary 4.6. Finally, we terminated the algorithm when $L = U$, solving the robust problem to optimality.

3.2.1 Analysis of Problem Solutions

To understand the effect of the budget of uncertainty Γ utilized by the decision-maker in the selection of the uncertainty set $\mathcal{B}$, we solved both instances of the newsvendor problem for values of Γ ranging from 0 to 50. Figures 3.1–3.4 summarize our findings.

Figures 3.1 and 3.3 show the worst-case cost of the two instances (i.e., the optimal objective value of problem (2.7)) as a function of Γ, which, as expected, increase as the solution becomes more conservative (the green curves). To assess the average performance of the robust solutions, we created a sample of 5000 realizations of the demands, using independent normal random variables with mean $\bar{b}_i$ and standard deviation $0.4 \cdot \bar{b}_i$ for each i. The resulting average costs of the robust solutions are depicted in Figures 3.1 and 3.3 (the red curves in Figures 3.1 and 3.3 and, on a different scale, in Figures 3.2 and 3.4; the blue error bars reflect the sample standard deviations). For both instances of the newsvendor problem, we observe from Figures 3.2 and 3.4 that the average cost first decreases with Γ, as incorporating a small amount of uncertainty in the model yields more robust solutions, reaches its minimum, and starts increasing with Γ as the solution becomes overly conservative for the typical demand realization. In Figure 3.2, the optimal trade-off is reached at $\Gamma = 5$, and the average cost of the corresponding robust solution achieves savings of 3.4% over the solution obtained for $\Gamma = 0$ (i.e., the solution targeted to satisfy the nominal demand $\bar{\mathbf{b}}$), while Figure 3.4 has an optimal trade-off at $\Gamma = 11$ and savings of 4.1%; both are consistent with the guidelines provided by Bertsimas and

Sim in [15], namely, that the budget of uncertainty should be of the order of $\sqrt{n}$ (here $\sqrt{50} \approx 7.1$).

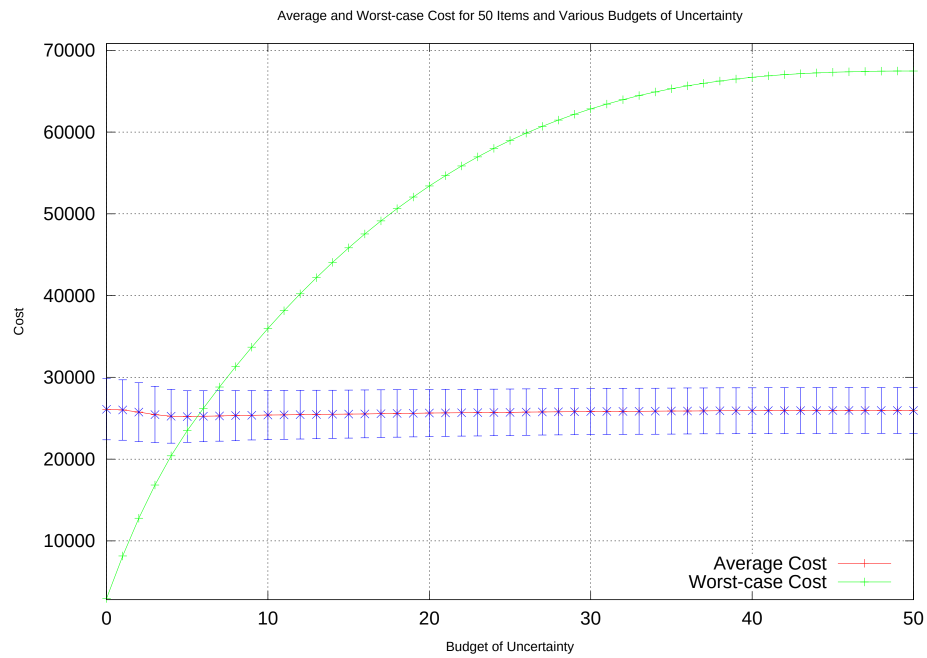


Figure 3.1: The impact of the budget of uncertainty on worst-case cost for Instance 1 of the multi-item newsvendor problem.

To strengthen our results, we found that the 99.96% confidence interval surrounding the mean of the differences (for each demand instance we subtracted the minimum Γ cost from the $\Gamma = 0$ cost and took the average over these differences) is far from containing zero for both Instance 1 and 2 (see Table 3.2.1 for the statistical information and confidence interval for each data instance), and thus we can say with high statistical significance, $p = 0.0004$, that the average cost at $\Gamma = 5$ for Instance 1 and $\Gamma = 11$ for Instance 2 is lower than $\Gamma = 0$, resulting in a savings in average cost. Note that for the first instance, the worst case for $\Gamma = 5$ corresponds to the situation where the demand for the last five items (items 46 to 50) is equal to its

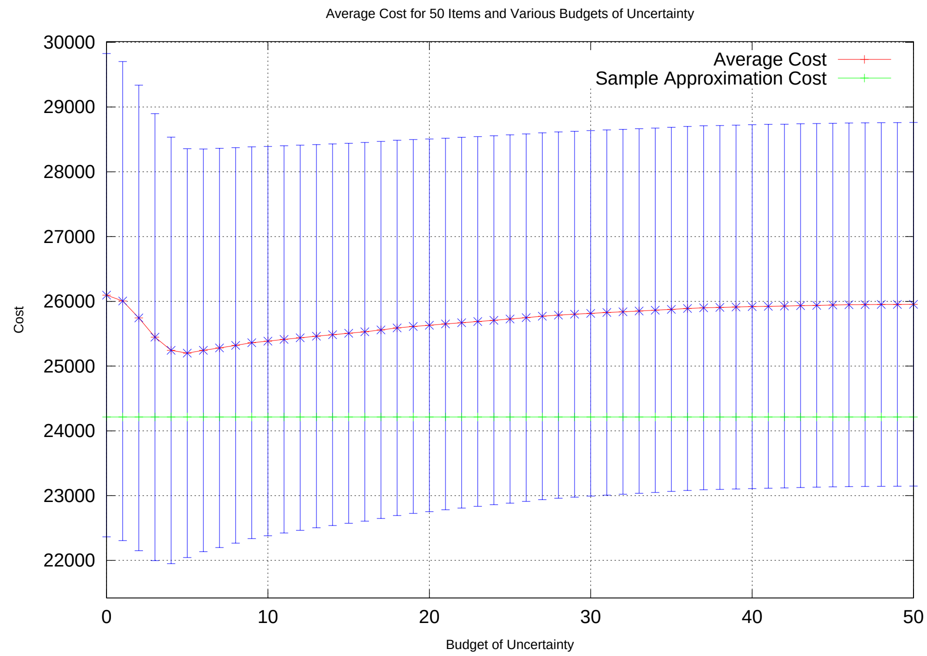


Figure 3.2: The impact of the budget of uncertainty on average cost for Instance 1 of the multi-item newsvendor problem.

Cost Data	Mean of Differences	Std Dev of Differences	99.96% Confidence Interval
Instance 1	895.54	2427.48	$[775.73, 1015.35]$
Instance 2	653.79	896.78	$[609.53, 698.05]$

Table 3.2: 99.96% confidence intervals for the mean of the differences for Instances 1 and 2.

highest value and demand for the other items is equal to its nominal value, which makes sense as the last five items have the largest shortage and holding penalties. In the second instance, the worst-case instance for $\Gamma = 11$ consists of demand for 11 products with mid-range penalties equal to its highest value.

In an attempt to compare the robust methodology to stochastic programming, we solved the sample average approximation (SAA) of the stochastic version of the newsvendor problem for both Instance 1 and 2 with the same normal demand sample that was used to assess the average performance of the robust solutions. Thus, the

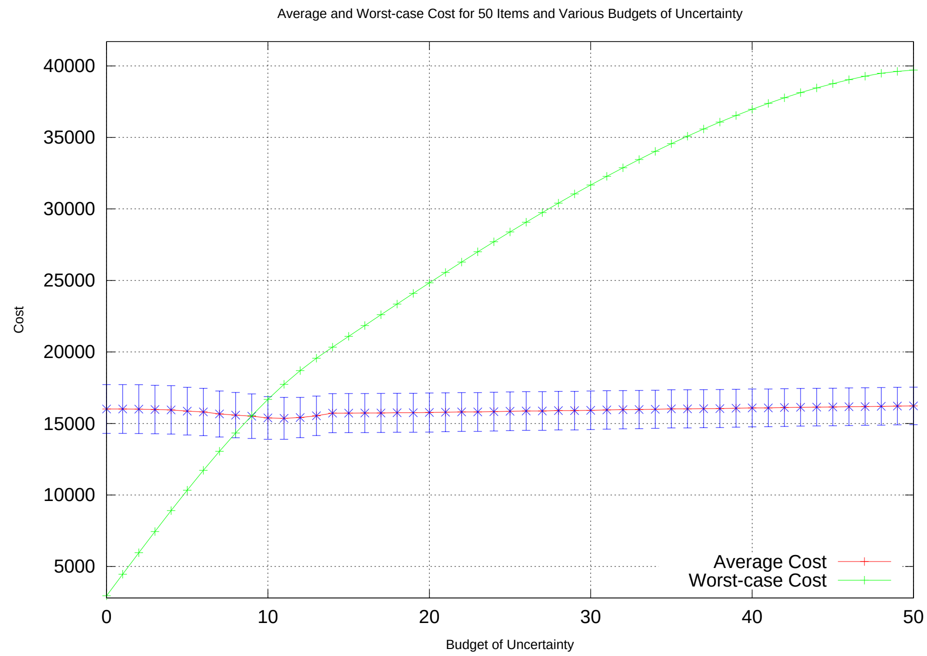


Figure 3.3: The impact of the budget of uncertainty on worst-case cost for Instance 2 of the multi-item newsvendor problem.

green lines in Figures 3.2 and 3.4 represent the cost of the sample average approximation ordering policy against the normal demand sample for Instance 1 and 2, respectively. As expected, the sample average approximation cost is lower for both Instance 1 and 2; however, the minimum average costs resulting from the robust methodology are only 4.1% and 1.9% larger than the sample average approximation cost for Instances 1 and 2, respectively. If the demands are coming from a distribution known to be normal with a known mean and standard deviation, then solving the stochastic approximation would provide a lower costing ordering policy, but if the distribution is unknown or uncertain prior to realization of the demands, then the robust methodology would provide a good ordering policy at a slightly higher cost.

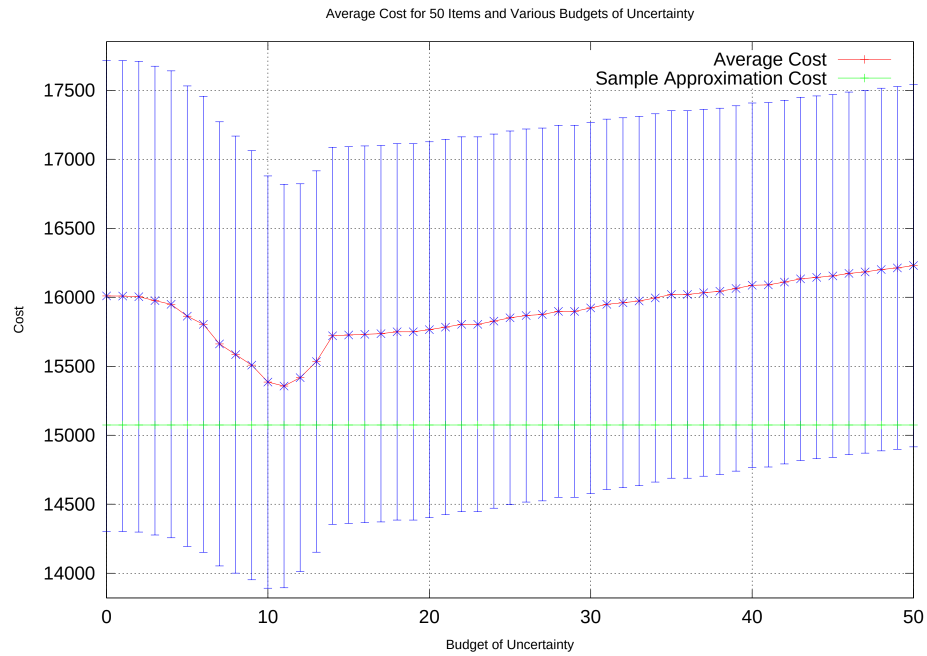


Figure 3.4: The impact of the budget of uncertainty on average cost for Instance 2 of the multi-item newsvendor problem.

3.2.2 Algorithmic Performance

Figures 3.5–3.8 illustrate the effect of the budget of uncertainty on both the number of iterations and the running time, in CPU seconds, of Algorithm 2.20. Neither the number of iterations nor the running time showed any particular dependence on Γ (although problems with very small and very large values of Γ appear easier to solve, due to relatively small numbers of extreme points of $\mathcal{B}$). The maximum number of iterations needed for either problem instance was 182, while the maximum running time was under two seconds.

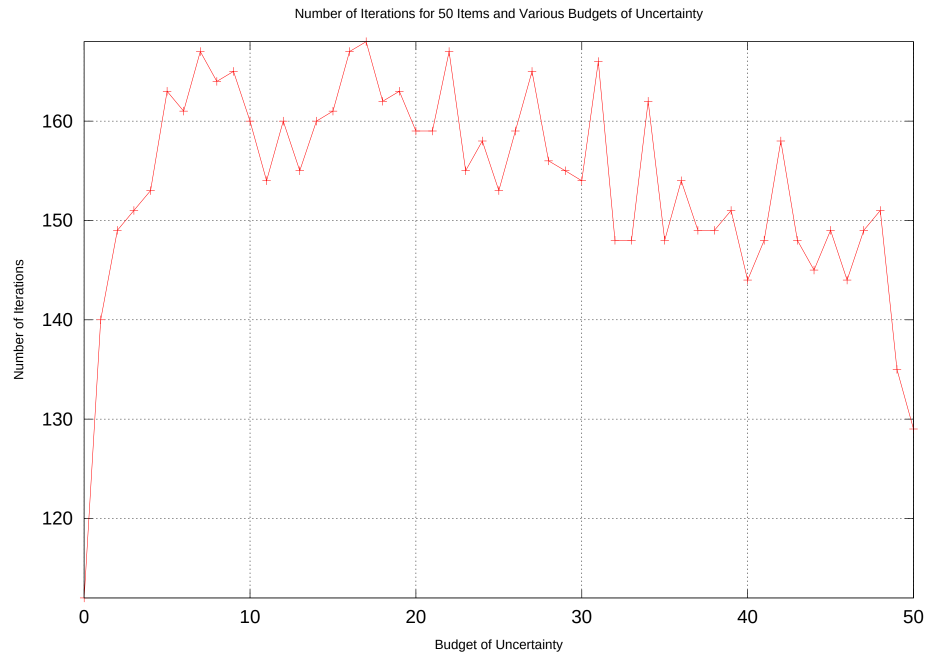


Figure 3.5: The impact of the budget of uncertainty on the number of iterations for Instance 1 of the multi-item newsvendor problem.

3.3 Additional Experiments and Computational Results for the Newsvendor Problem

3.3.1 Kelley's Method versus ACCPM and Subgradient Algorithm for the Newsvendor Problem

Kelley's method (Algorithm 2.15) was chosen over both the ACCPM (Algorithm 2.17) and the subgradient algorithm (Algorithm 2.18) because of the low number of iterations needed for convergence and for the speedy running time. Both the ACCPM and subgradient algorithm require many more iterations and a longer running time (CPU seconds) than Kelley's method. Table 3.3 presents our performance results for the three algorithms on the newsvendor problem (Instance 1) for $\Gamma = 0$. In the ACCPM, the termination criteria were $U - L < \epsilon$ ($\epsilon = 0.1$) or the number of iterations equaled 600, whichever came first. Similarly, in the subgradient method,

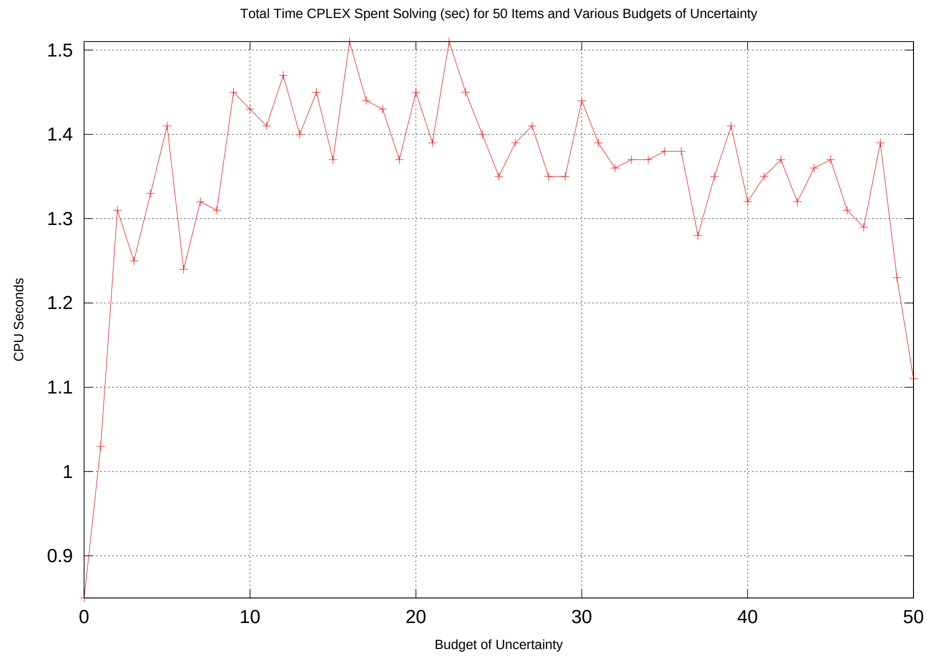


Figure 3.6: The impact of the budget of uncertainty on the run time (sec) for Instance 1 of the multi-item newsvendor problem.

the termination criteria were $U - L < \epsilon$ ($\epsilon = 0.1$) or the number of iterations equaled 10,000, whichever came first. As Table 3.3 shows, both the ACCPM and subgradient algorithm terminated when reaching the maximum number of iterations allowed.

$\Gamma = 0$	Number of Iterations	Running Time (CPU seconds)
Kelley's method	112	0.85
ACCPM	600	140,500.4
Subgradient	10,000	5362.6

Table 3.3: Number of iterations and running time (sec) for Algorithms 2.15, 2.17, and 2.18 for $\Gamma = 0$.

We make no claims on the efficiency of the implementation of either the ACCPM or subgradient algorithm and we are aware that efficient software implementations of both are available, which could have been used to improve the efficiency and performance of the ACCPM and subgradient algorithm; however, we'd like to offer

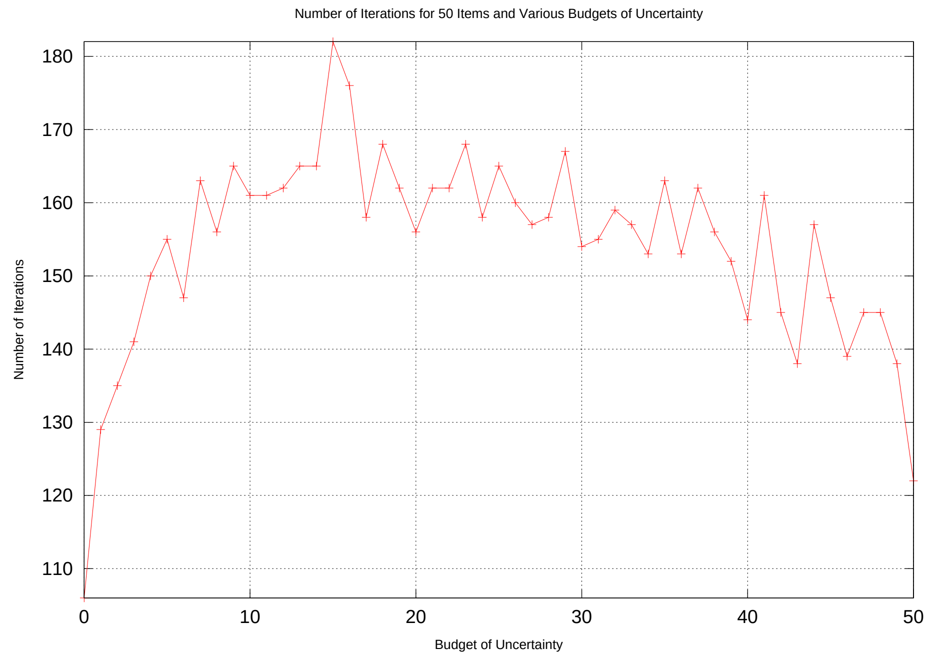


Figure 3.7: The impact of the budget of uncertainty on the number of iterations for Instance 2 of the multi-item newsvendor problem.

up the following analysis, which is independent of our implementation and stands even if we improve the efficiency of our implementations, as reasons for not pursuing these two algorithms in this setting. If we had used available efficient software in our implementation of the ACCPM, we know the running time per iteration would decrease by some unknown quantity. Suppose the efficient software would have decreased the ACCPM run time per iteration to be equal to that of Kelley's method (solving a relatively small LP). Notice the ACCPM required more than 600 iterations to decrease the bound gap to within 0.1, while Kelley's method needed only 112 iterations to decrease the bound gap to zero. Thus, even if the time per iteration of the ACCPM was equal to that of Kelley's method, the larger number of iterations required for convergence would still make the ACCPM worse compared to

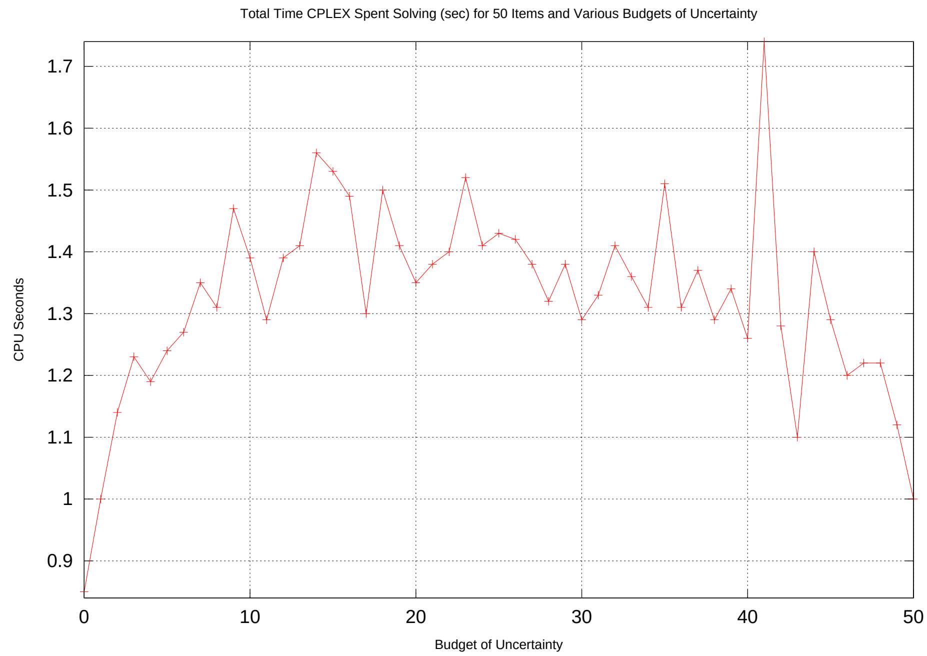


Figure 3.8: The impact of the budget of uncertainty on the run time (sec) for Instance 2 of the multi-item newsvendor problem.

Kelley's method for robust linear programs with simple recourse and right-hand side uncertainty.

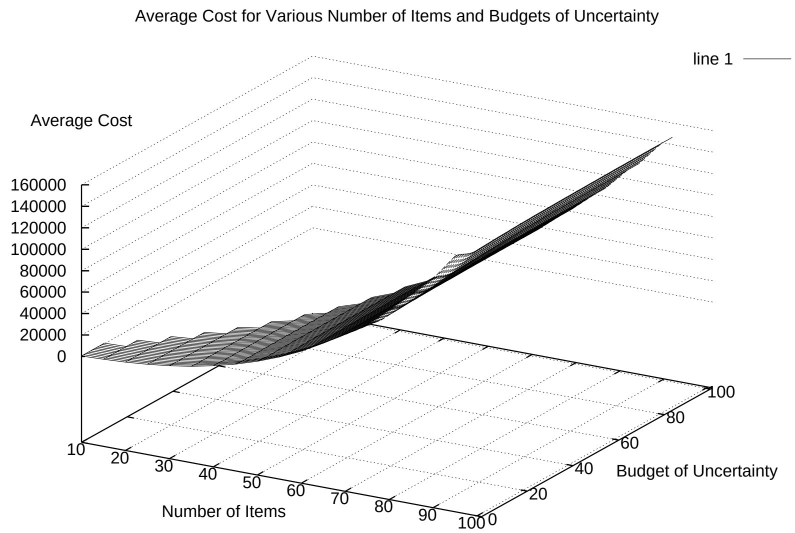
When the maximum number of iterations allowed for the subgradient algorithm was increased from 10,000 to 100,000, the bound gap was still greater than 0.1 when 100,000 iterations had executed. The subgradient algorithm would find a good approximate solution within a second or two (reducing the bound gap to 100 or 200), but would spend the remaining time tightening the bound and trying to find the optimal solution. There might be some potential for a hybrid algorithm that starts with the subgradient algorithm to find a good approximate solution and then switches to Kelley's method to find the optimal solution. However, the difficulty the subgradient algorithm had in narrowing in on the optimal solution (requiring more than 1,000 times the number of iterations of Kelley's method) made Kelley's method a more suitable choice for robust linear programs with simple recourse and right-hand side uncertainty.

3.3.2 Further Analysis of Kelley's Method for the Newsvendor Problem

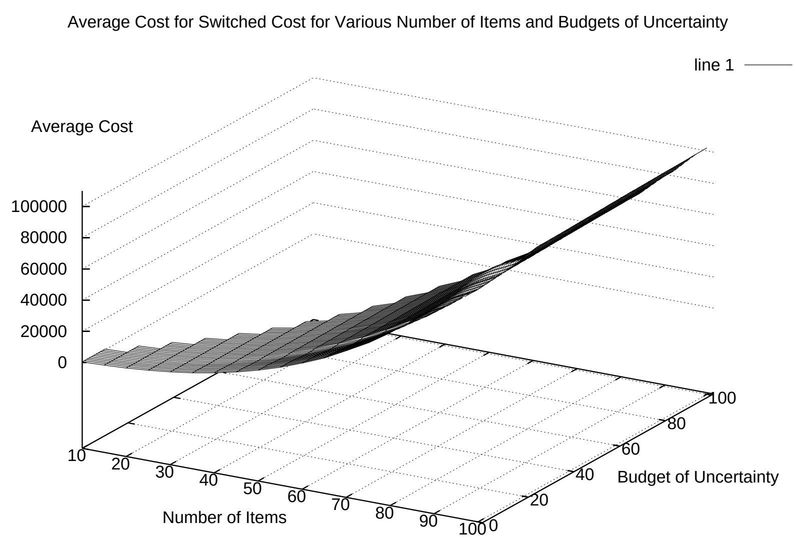
In section 3.2, the data instances we looked at each had $n = 50$ items, while we varied $\Gamma \in [0, 50]$. Here we consider both smaller and larger values of n to see how the size of the problem, as well as the budget of uncertainty, affects the computational performance of Algorithm 2.20 (Kelley's method). Figures 3.9–3.12 display average cost, worst-case cost, number of iterations, and running time (in seconds) as n increases from 10 up to 100 by increments of 5 for both instances of the newsvendor problem (see Table 3.1). For each instance in Table 3.1 and for each value of n, we used Algorithm 2.20 to solve the newsvendor problem for each $\Gamma \in [0, n]$. There is an associated worst-case cost, number of iterations performed, and running time (in CPU seconds) for each (n, Γ)-pair. The average cost is estimated by using the optimal solution found for each (n, Γ)-pair and sampling 500 demands for each (n, Γ)-pair (with the sampling as described in section 3.2).

Figures 3.9(a) and 3.9(b) show that the average cost follows a gently sloping convex surface that increases as the number of items increase and the budget of uncertainty increases. While the curves look very similar, notice that the two problem instances of the newsvendor problem, Instance 1, shown in Figure 3.9(a), results in a much larger average cost than that of Instance 2, shown in Figure 3.9(b). Figures 3.10(a) and 3.10(b) show that, for a fixed n, worst-case cost increases as more demands are allowed to deviate from their nominal values for a fixed value of n and resembles the concave curves we saw in Figures 3.1–3.4.

Figures 3.11(a) and 3.11(b) show a rather linear surface, indicating that as the

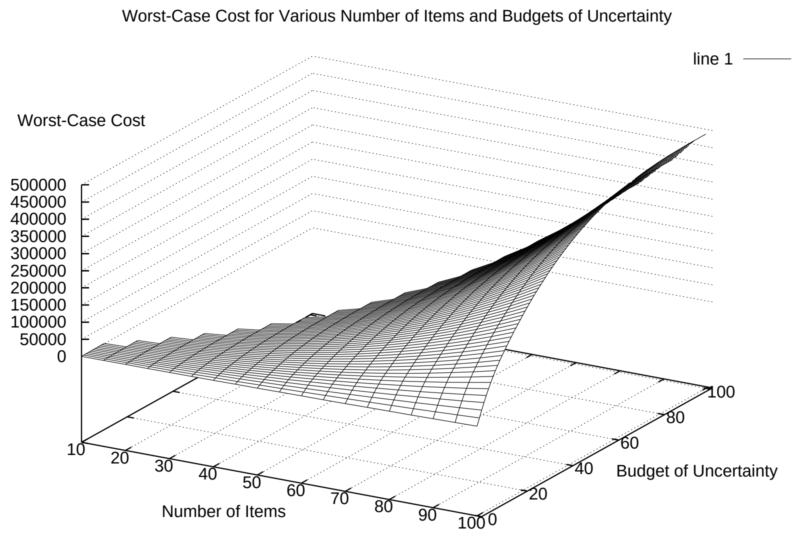


(a) Instance 1: Average Cost versus (n, Γ).

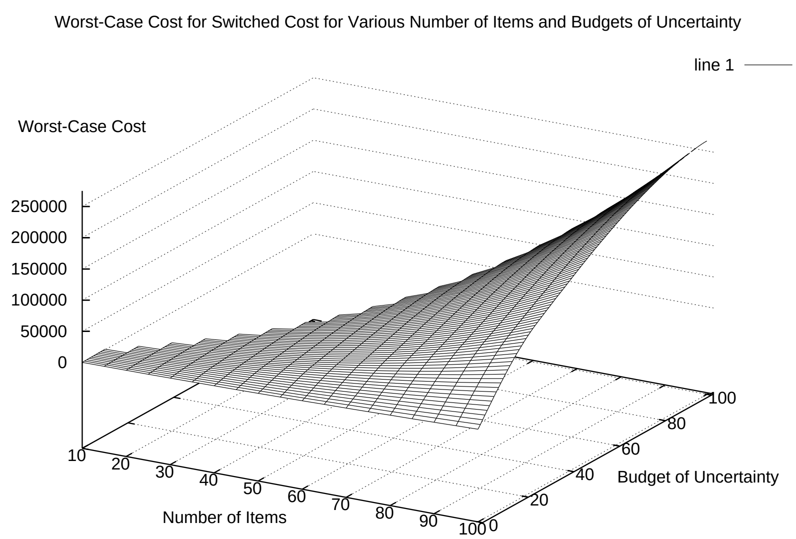


(b) Instance 2: Average Cost versus (n, Γ).

Figure 3.9: The impact of the number of items and the budget of uncertainty on average cost for two instances of the multi-item newsvendor problem.



(a) Instance 1: Worst-Case Cost versus (n, Γ).

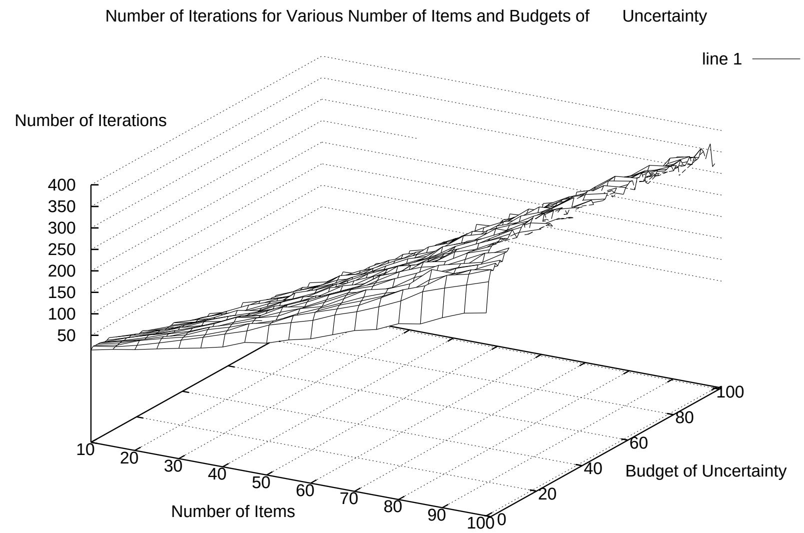


(b) Instance 2: Worst-Case Cost versus (n, Γ).

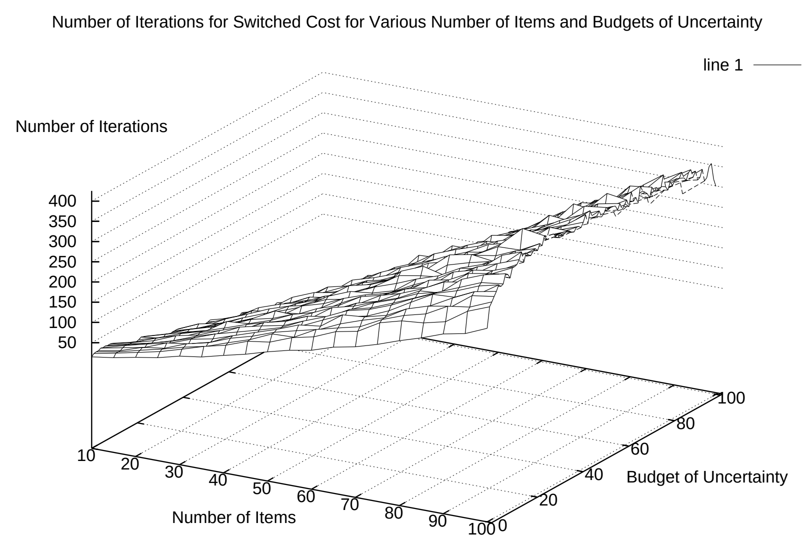
Figure 3.10: The impact of the number of items and the budget of uncertainty on worst-case cost for two instances of the multi-item newsvendor problem.

number of items increases, the number of iterations increases. Although the number of iterations increases with n, for a fixed number of items, no dependence is exhibited between the number of iterations and the budget of uncertainty for either instance of the problem; the same result as shown in Figures 3.5–3.8.

The running time (in seconds) is shown in Figures 3.12(a) and 3.12(b). It exhibits a faster-than-linear growth in running time as the number of items increase. Again, as seen in Figures 3.5–3.8, no dependence between the running time and Γ is exhibited for a fixed n. Next, we considered whether the increase in the running time is due to the increase in the number of iterations required, or the increased work per iteration needed, as the algorithm is working with larger vectors and matrices as n increases. To address this concern, we plotted the ratio between running time and the number of iterations required for each (n, Γ)-pair and each problem instance. Figures 3.13(a) and 3.13(b) show that the running time is increasing faster than the number of iterations required as n increases, resulting in the conclusion that the running time is increasing due to the increased work per iteration. It should be noted that we did not make any special efforts to implement the individual steps of our algorithm efficiently (e.g., we did not take advantage of the fact that the linear program being solved at each iteration differs from the one solved in the previous iteration by one additional constraint, and hence is likely easily handled by the dual simplex algorithm). Such savings could potentially improve the running time of the algorithm, although it is unlikely that they will qualitatively change the plots in Figures 3.13(a) and 3.13(b).

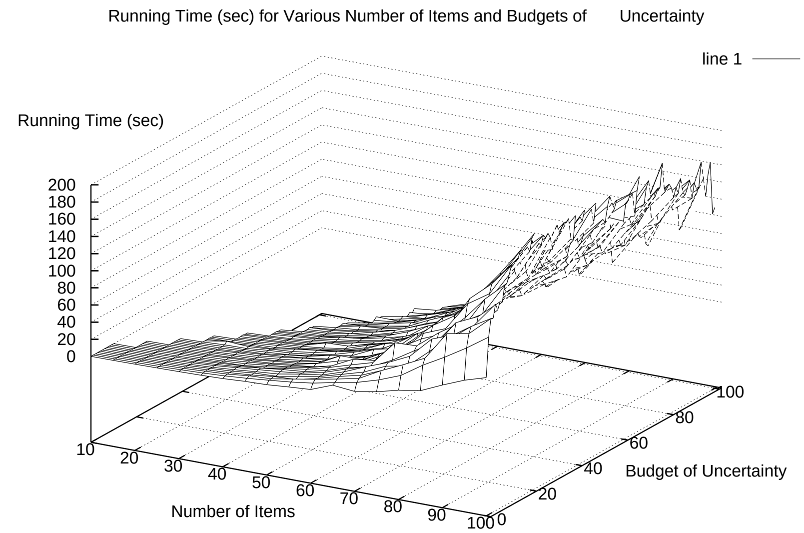


(a) Instance 1: Iteration Count versus (n, Γ).

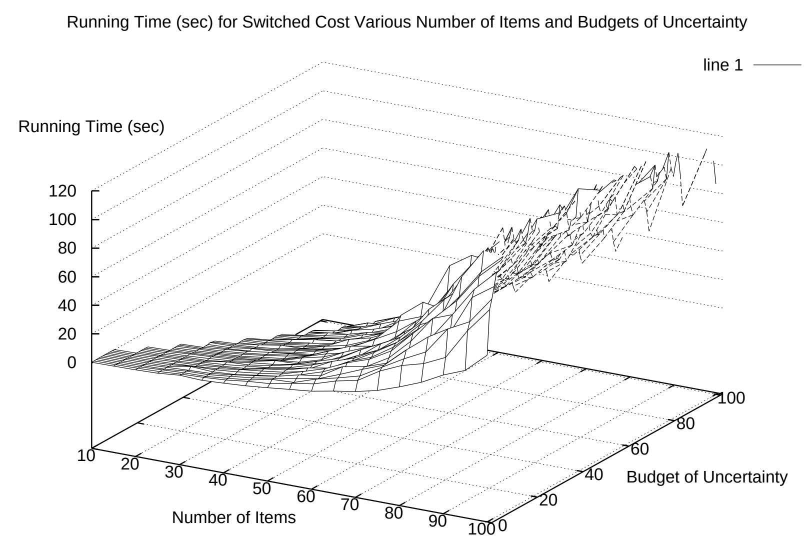


(b) Instance 2: Iteration Count versus (n, Γ).

Figure 3.11: The impact of the number of items and the budget of uncertainty on the number of iterations for two instances of the multi-item newsvendor problem.

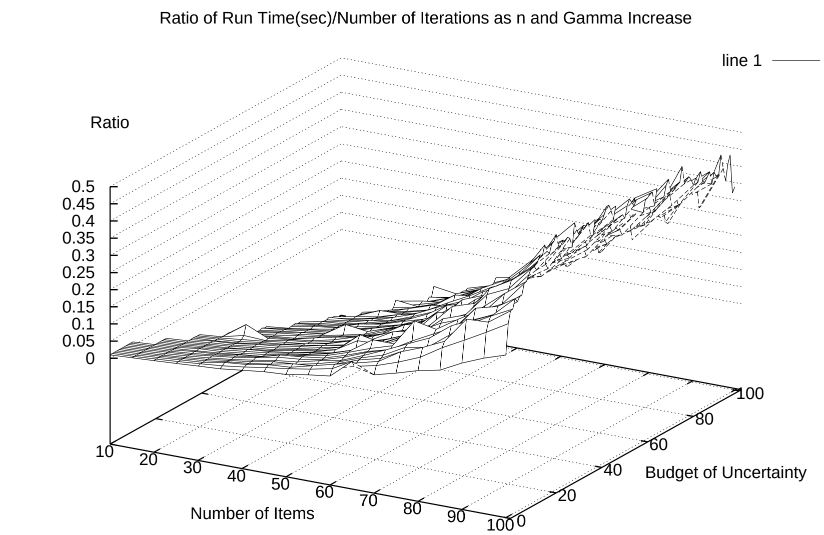


(a) Instance 1: Running Time (sec) versus (n, Γ).

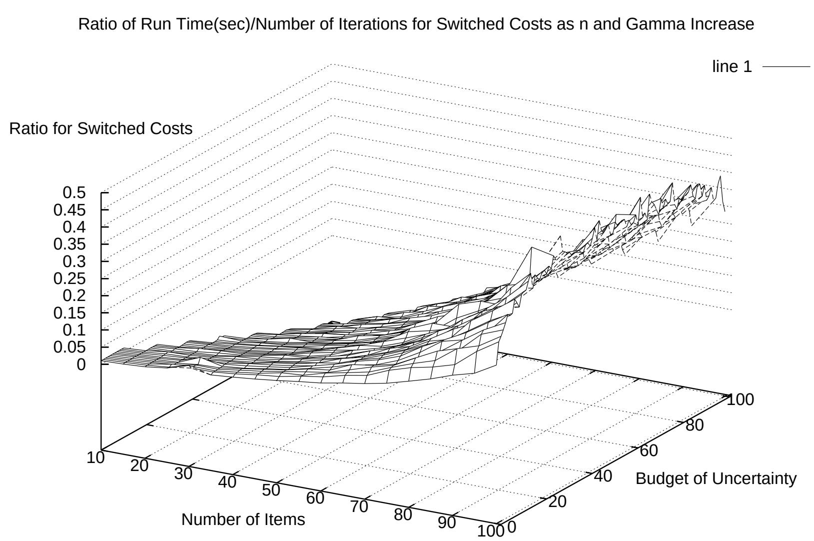


(b) Instance 2: Running Time (sec) versus (n, Γ).

Figure 3.12: The impact of the number of items and the budget of uncertainty on the run time (sec) for two instances of the multi-item newsvendor problem.



(a) Instance 1: Ratio of Running Time and Number of Iterations versus (n, Γ).



(b) Instance 2: Ratio of Running Time and Number of Iterations versus (n, Γ).

Figure 3.13: The impact of the number of items and the budget of uncertainty on the ratio of run time (sec) and number of iterations for two instances of the multi-item newsvendor problem.

CHAPTER IV

Analysis of Robust Linear Optimization Problems with General Recourse and Computational Experiments

In this chapter, we present a variety of experimental results when using Kelley's Method to solve a production planning problem with general recourse including an analysis of problem solutions and performance results of Kelley's Method. [1]

4.1 Analysis of Robust Linear Programs with General Recourse

In this section, we return to the analysis of the robust linear program with general recourse function, problem (2.2). Without any assumptions on the structure of the recourse matrix $\mathbf{B}$, evaluation of $\mathcal{Q}(\mathbf{x}) = \max_{\mathbf{b}\in\mathcal{B},\mathbf{p}:\mathbf{B}^T\mathbf{p}\leq\mathbf{d}}(\mathbf{b}-\mathbf{A}\mathbf{x})^T\mathbf{p}$ can no longer be done in closed form, as was the case with simple recourse in Chapter III. We present a general approach for computing the value of $\mathcal{Q}(\mathbf{x})$, along with the corresponding worst-case value of $\mathbf{b}$ and dual recourse variable $\mathbf{p}$, via a mixed-integer programming problem.

[1] Some of the analysis and experimental results presented here can also be found in Thiele et al. [63].

4.1.1 Computing $\mathcal{Q}(\mathbf{x})$ for Robust Linear Programs with General Recourse

Theorem 4.1. *Given* $\mathbf{x}$, $\mathcal{Q}(\mathbf{x}) = \max_{\mathbf{b}\in\mathcal{B}} Q(\mathbf{x},\mathbf{b})$ *where* $Q(\mathbf{x},\mathbf{b})$ *is given by equation* (2.6) *and* $\mathcal{B}$ *given by equations* (2.8)–(2.9) *can be computed by*

$$
\begin{aligned}
\mathcal{Q}(\mathbf{x}) = \max_{\mathbf{p}^+,\mathbf{p}^-,\mathbf{q}^+,\mathbf{q}^-,\mathbf{r}^+,\mathbf{r}^-} \quad & (\overline{\mathbf{b}} - \mathbf{A}\mathbf{x})^T(\mathbf{p}^+ - \mathbf{p}^-) + \widehat{\mathbf{b}}^T(\mathbf{q}^+ + \mathbf{q}^-) \\
s.t. \quad & \mathbf{B}^T(\mathbf{p}^+ - \mathbf{p}^-) \le \mathbf{d}, \\
& \mathbf{0} \le \mathbf{q}^+ \le \mathbf{p}^+, \\
& \mathbf{0} \le \mathbf{q}^- \le \mathbf{p}^-, \\
& \mathbf{q}^+ \le M\mathbf{r}^+, \\
& \mathbf{q}^- \le M\mathbf{r}^-, \\
& \mathbf{e}^T(\mathbf{r}^+ + \mathbf{r}^-) \le \Gamma, \\
& \mathbf{r}^+ + \mathbf{r}^- \le \mathbf{e}, \\
& \mathbf{r}^+, \mathbf{r}^- \in \{0,1\}^m, \\
& \mathbf{p}^+, \mathbf{p}^- \ge \mathbf{0},
\end{aligned} \tag{4.2}
$$

where $\mathbf{e}$ *is the vector of all ones and* M *is a sufficiently large positive number.*

Proof. Let $\mathbf{p} = \mathbf{p}^+ - \mathbf{p}^-$, with $\mathbf{p}^+, \mathbf{p}^- \ge \mathbf{0}$. Recalling the definition of $\mathcal{B}$, $\mathcal{Q}(\mathbf{x}) = \max_{\mathbf{b}\in\mathcal{B},\mathbf{p}:\mathbf{B}^T\mathbf{p}\le\mathbf{d}} (\mathbf{b} - \mathbf{A}\mathbf{x})^T\mathbf{p}$ can be rewritten as

$$
\begin{aligned}
\mathcal{Q}(\mathbf{x}) = \max_{\mathbf{p}^+,\mathbf{p}^-} \quad & (\overline{\mathbf{b}} - \mathbf{A}\mathbf{x})^T(\mathbf{p}^+ - \mathbf{p}^-) + \max_{\mathbf{z}\in\mathcal{Z}} \sum_{i=1}^m \widehat{b}_i(p_i^+ - p_i^-)z_i \\
\text{s.t.} \quad & \mathbf{B}^T(\mathbf{p}^+ - \mathbf{p}^-) \le \mathbf{d} \\
& \mathbf{p}^+, \mathbf{p}^- \ge \mathbf{0}
\end{aligned} \tag{4.3}
$$

where $\mathcal{Z}$ was defined in equation (2.9). Note that for a given $\mathbf{p}^+$ and $\mathbf{p}^-$, the inner maximization problem of (4.3) is a linear program in variables $\mathbf{z}$ and its optimal solution can be found at one of the extreme points of the set $\mathcal{Z}$, which have the form $z_i \in \{-1,0,1\}\ \forall i$ and $\sum_{i=1}^m |z_i| = \Gamma$. Therefore, we can rewrite the inner maximization problem of (4.3) as the following integer program in variables $\mathbf{r}^+, \mathbf{r}^-$:

$$
\begin{aligned}
\max_{\mathbf{r}^+,\mathbf{r}^-} \quad & \sum_{i=1}^{m} \widehat{b}_i (p_i^+ - p_i^-)(r_i^+ - r_i^-) \\
\text{s.t.} \quad & \mathbf{e}^T(\mathbf{r}^+ + \mathbf{r}^-) \leq \Gamma \\
& \mathbf{r}^+ + \mathbf{r}^- \leq \mathbf{e} \\
& \mathbf{r}^+, \mathbf{r}^- \in \{0,1\}^m.
\end{aligned} \tag{4.4}
$$

Substituting problem (4.4) into problem (4.3), we obtain the following equivalent formulation:

$$
\begin{aligned}
\mathcal{Q}(\mathbf{x}) = \max_{\mathbf{p}^\pm,\mathbf{r}^\pm} \quad & (\overline{\mathbf{b}} - \mathbf{A}\mathbf{x})^T(\mathbf{p}^+ - \mathbf{p}^-) + \sum_{i=1}^{m} \widehat{b}_i (p_i^+ - p_i^-)(r_i^+ - r_i^-) \\
\text{s.t.} \quad & \mathbf{B}^T(\mathbf{p}^+ - \mathbf{p}^-) \leq \mathbf{d}, \\
& \mathbf{e}^T(\mathbf{r}^+ + \mathbf{r}^-) \leq \Gamma, \\
& \mathbf{r}^+ + \mathbf{r}^- \leq \mathbf{e}, \\
& \mathbf{r}^+,\ \mathbf{r}^- \in \{0,1\}^m, \\
& \mathbf{p}^+,\ \mathbf{p}^- \geq \mathbf{0}.
\end{aligned} \tag{4.5}
$$

Finally, to linearize the objective function of problem (4.5), we introduce variables $q_i^+ = p_i^+ r_i^+$ and $q_i^- = p_i^- r_i^-$ $\forall i$ (note we can assume $p_i^+ r_i^- = p_i^- r_i^+ = 0$ $\forall i$ without loss of generality). Making this substitution in problem (4.5) and adding appropriate forcing constraints results in problem (4.2). $\square$

Corollary 4.6. *Given* $\mathbf{x}$, *let* $(\mathbf{p}^\pm, \mathbf{r}^\pm, \mathbf{q}^\pm)$ *solve problem* (4.2). *Then the corresponding worst-case value of* $\mathbf{b}$ *can be determined as follows:*

$$b_i = \overline{b}_i + \widehat{b}_i (r_i^+ - r_i^-),\ \ i = 1, \ldots, m,$$

and the corresponding dual recourse variable $\mathbf{p}$ *can be determined as* $\mathbf{p} = \mathbf{p}^+ - \mathbf{p}^-$.

The subgradient of $\mathcal{Q}(\mathbf{x})$ can now be computed as in Lemma 2.19.

4.2 Computational Results: Production Planning Problem

Here, we consider a production planning example where the demand is uncertain, but must be met. Once demand has been revealed, the decision-maker has the option to buy additional raw materials at a higher cost and re-run the production process, so that demand for all products is satisfied. The goal is to minimize the ordering costs of the raw materials and the production costs of the finished products, as well as the inventory (or disposal) costs on the materials and products remaining at the end of the time period. We define the following notation:

m : the number of raw materials,

n : the number of finished products,

$\mathbf{c}$: the first-stage unit cost of the raw materials,

$\mathbf{d}$: the second-stage unit cost of the raw materials,

$\mathbf{f}$: the first-stage unit production cost,

$\mathbf{g}$: the second-stage unit production cost,

$\mathbf{h}$: the unit inventory cost of unused raw materials,

$\mathbf{k}$: the unit inventory cost of unsold finished products,

$\mathbf{A}$: the productivity matrix,

$\mathbf{b}$: the demand for the finished products,

$\mathbf{x}$: the raw materials purchased in the first stage,

$\mathbf{y}$: the raw materials purchased in the second stage,

$\mathbf{u}$: the products produced in the first stage,

$\mathbf{v}$: the products produced in the second stage.

We assume that all coefficients of the matrix $\mathbf{A}$ are nonnegative as are all costs:

$\mathbf{c}$, $\mathbf{d}$, $\mathbf{f}$, $\mathbf{g}$, $\mathbf{h}$, and $\mathbf{k}$. The deterministic problem can be formulated as:

$$
\begin{aligned}
\min_{\mathbf{x},\mathbf{y},\mathbf{u},\mathbf{v}} \quad & \mathbf{c}^T\mathbf{x} + \mathbf{d}^T\mathbf{y} + \mathbf{f}^T\mathbf{u} + \mathbf{g}^T\mathbf{v} + \mathbf{h}^T(\mathbf{x} + \mathbf{y} - \mathbf{A}(\mathbf{u} + \mathbf{v})) + \mathbf{k}^T(\mathbf{u} + \mathbf{v} - \mathbf{b}) \\
\text{s.t.} \quad & \mathbf{A}\mathbf{u} \le \mathbf{x}, \\
& \mathbf{A}\mathbf{u} + \mathbf{A}\mathbf{v} \le \mathbf{x} + \mathbf{y}, \\
& \mathbf{u} + \mathbf{v} \ge \mathbf{b}, \\
& \mathbf{x},\ \mathbf{y},\ \mathbf{u},\ \mathbf{v} \ge \mathbf{0}.
\end{aligned} \tag{4.7}
$$

Note that, according to the formulation, raw materials can be purchased in the first stage of the time period and used in production in the second stage.

4.2.1 Analysis of Production Planning Problem

The robust production planning problem is as follows:

$$
\begin{aligned}
\min_{\mathbf{x},\mathbf{u}} \quad & \mathbf{c}^T\mathbf{x} + \mathbf{f}^T\mathbf{u} + \max_{\mathbf{b}\in\mathcal{B}} Q(\mathbf{x},\mathbf{u},\mathbf{b}) \\
\text{s.t.} \quad & \mathbf{A}\mathbf{u} \le \mathbf{x} \\
& \mathbf{x},\ \mathbf{u} \ge \mathbf{0},
\end{aligned} \tag{4.8}
$$

where the recourse problem, $Q(\mathbf{x},\mathbf{u},\mathbf{b})$, is given by:

$$
\begin{aligned}
Q(\mathbf{x},\mathbf{u},\mathbf{b}) = \bar{c} - \mathbf{k}^T\mathbf{b} + \min_{\mathbf{y},\mathbf{v}} \quad & (\mathbf{d} + \mathbf{h})^T\mathbf{y} + (\mathbf{g} - \mathbf{A}^T\mathbf{h} + \mathbf{k})^T\mathbf{v}, \\
\text{s.t.} \quad & \mathbf{y} - \mathbf{A}\mathbf{v} \ge \mathbf{A}\mathbf{u} - \mathbf{x}, \\
& \mathbf{v} \ge \mathbf{b} - \mathbf{u}, \\
& \mathbf{y}, \mathbf{v} \ge \mathbf{0},
\end{aligned} \tag{4.9}
$$

and $\bar{c}(\mathbf{x},\mathbf{u}) = \mathbf{h}^T\mathbf{x} - \mathbf{h}^T\mathbf{A}\mathbf{u} + \mathbf{k}^T\mathbf{u}$ is a constant. Since this form of the recourse function is slightly different from expression (2.6) in that the constraints are in inequality form, we begin by deriving a modification of the mixed-integer program of Theorem 4.1.

The dual of the recourse problem, problem (4.9), is:

$$
\begin{aligned}
Q(\mathbf{x},\mathbf{u},\mathbf{b}) = \bar{c}(\mathbf{x},\mathbf{u}) - \mathbf{k}^T\mathbf{b} + \max_{\mathbf{q},\mathbf{p}} \quad & (\mathbf{A}\mathbf{u}-\mathbf{x})^T\mathbf{q} + (\mathbf{b}-\mathbf{u})^T\mathbf{p} \\
\text{s.t.} \quad & \mathbf{0} \le \mathbf{q} \le \mathbf{d} + \mathbf{h} \\
& \mathbf{0} \le \mathbf{p} \le \mathbf{A}^T\mathbf{q} + \mathbf{g} - \mathbf{A}^T\mathbf{h} + \mathbf{k},
\end{aligned}
\tag{4.10}
$$

To obtain $\mathcal{Q}(\mathbf{x},\mathbf{u}) = \max_{\mathbf{b}\in\mathcal{B}} Q(\mathbf{x},\mathbf{u},\mathbf{b})$, where $Q(\mathbf{x},\mathbf{u},\mathbf{b})$ is defined by problem (4.10), we can write:

(4.11)

$$
\begin{aligned}
\mathcal{Q}(\mathbf{x},\mathbf{u}) = \bar{c}(\mathbf{x},\mathbf{u}) - \mathbf{k}^T\mathbf{b} + & \\
\max_{\mathbf{q},\mathbf{p}} \quad & \Big((\mathbf{A}\mathbf{u}-\mathbf{x})^T\mathbf{q} + (\overline{\mathbf{b}}-\mathbf{u})^T\mathbf{p} + \max_{\mathbf{z}\in\mathcal{Z}} \sum_{i=1}^{n} \big(\widehat{b}_i(p_i - k_i)z_i\big)\Big) \\
\text{s.t.} \quad & \mathbf{0} \le \mathbf{q} \le \mathbf{d} + \mathbf{h} \\
& \mathbf{0} \le \mathbf{p} \le \mathbf{A}^T\mathbf{q} + \mathbf{g} - \mathbf{A}^T\mathbf{h} + \mathbf{k},
\end{aligned}
$$

where $\mathcal{Z}$ was defined in equation (2.9). Applying the same logic as in Theorem 4.1, for a given $\mathbf{q}$ and $\mathbf{p}$, we can rewrite the inner maximization problem of problem (4.11) as an integer programming problem in variables $\mathbf{r}^+$ and $\mathbf{r}^-$:

$$
\begin{aligned}
\max_{\mathbf{r}^+,\mathbf{r}^-} \quad & \sum_{i=1}^{m} \widehat{b}_i(p_i - k_i)(r_i^+ - r_i^-) \\
\text{s.t.} \quad & \mathbf{e}^T(\mathbf{r}^+ + \mathbf{r}^-) \le \Gamma \\
& \mathbf{r}^+ + \mathbf{r}^- \le \mathbf{e} \\
& \mathbf{r}^+, \mathbf{r}^- \in \{0,1\}^m.
\end{aligned}
\tag{4.12}
$$

Substituting problem (4.12) into problem (4.11), we obtain:

(4.13)

$$\mathcal{Q}(\mathbf{x},\mathbf{u}) = \bar{c}(\mathbf{x},\mathbf{u}) - \mathbf{k}^T\mathbf{b} +$$

$$\begin{aligned}
\max_{\mathbf{q},\mathbf{p},\mathbf{r}^{\pm}} \quad & (\mathbf{A}\mathbf{u}-\mathbf{x})^T\mathbf{q} + (\overline{\mathbf{b}}-\mathbf{u}) + \sum_{i=1}^{m}\Big(\widehat{b}_i(p_i - k_i)(r_i^+ - r_i^-)\Big) \\
\text{s.t.} \quad & \mathbf{0} \le \mathbf{q} \le \mathbf{d}+\mathbf{h} \\
& \mathbf{0} \le \mathbf{p} \le \mathbf{A}^T\mathbf{q} + \mathbf{g} - \mathbf{A}^T\mathbf{h} + \mathbf{k} \\
& \mathbf{e}^T(\mathbf{r}^+ + \mathbf{r}^-) \le \Gamma \\
& \mathbf{r}^+ + \mathbf{r}^- \le \mathbf{e} \\
& \mathbf{r}^+,\ \mathbf{r}^- \in \{0,1\}^m
\end{aligned}$$

To remove the nonlinearity from the objective function, we introduce variables $s_i^+ = p_i r_i^+$ and $s_i^- = p_i r_i^-$ $\forall i$. Substituting these definitions into problem (4.13) we can conclude that we can calculate the value of $\mathcal{Q}(\mathbf{x},\mathbf{u})$ by solving the following mixed-integer program:

(4.14)

$$\bar{c}(\mathbf{x},\mathbf{u}) - \mathbf{k}^T\mathbf{b} +$$

$$\max_{\mathbf{q},\mathbf{p},\mathbf{s}^{\pm},\mathbf{r}^{\pm}} (\mathbf{A}\mathbf{u}-\mathbf{x})^T\mathbf{q} + (\overline{\mathbf{b}}-\mathbf{u})^T\mathbf{p} + \widehat{\mathbf{b}}^T(\mathbf{s}^+ - \mathbf{s}^-) - \sum_{i=1}^{n} k_i\widehat{b}_i(r_i^+ - r_i^-)$$

$$\begin{aligned}
\text{s.t.}\quad & \mathbf{0} \le \mathbf{q} \le \mathbf{d}+\mathbf{h} \\
& \mathbf{0} \le \mathbf{p} \le \mathbf{A}^T\mathbf{q} + \mathbf{g} - \mathbf{A}^T\mathbf{h} + \mathbf{k} \\
& \mathbf{0} \le \mathbf{s}^+ \le \mathbf{p} \\
& \mathbf{0} \le \mathbf{s}^- \le \mathbf{p} \\
& \mathbf{s}^+ \le M\mathbf{r}^+ \\
& \mathbf{s}^- \le M\mathbf{r}^- \\
& \mathbf{e}^T(\mathbf{r}^+ + \mathbf{r}^-) \le \Gamma \\
& \mathbf{r}^+ + \mathbf{r}^- \le \mathbf{e} \\
& \mathbf{r}^+, \mathbf{r}^- \in \{0,1\}^n,
\end{aligned}$$

where $\mathbf{e}$ is the vector of all ones and M is a sufficiently large positive number.

Lastly, given $(\mathbf{x},\mathbf{u})$, let $(\mathbf{p},\mathbf{q},\mathbf{r}^{\pm},\mathbf{s}^{\pm})$ solve problem (4.14). The corresponding dual recourse vector is $(\mathbf{p},\mathbf{q})$, and the corresponding worst-case value of $\mathbf{b}$ can be determined as $b_i = \overline{b}_i + \widehat{b}_i(r_i^+ - r_i^-),\ i = 1,\dots,m$. The subgradient of $\mathbf{c}^T\mathbf{x} + \mathbf{f}^T\mathbf{u} + \max_{\mathbf{b}\in\mathcal{B}} Q(\mathbf{x},\mathbf{u},\mathbf{b})$ can now be computed as in Lemma 2.19 and is equal to:

$$\begin{bmatrix} (\mathbf{c}-\mathbf{q}+\mathbf{h}) \\ (\mathbf{f}+\mathbf{A}^T\mathbf{q}-\mathbf{p}-\mathbf{A}^T\mathbf{h}+\mathbf{k}). \end{bmatrix} \tag{4.15}$$

4.2.2 Computational Results

We considered an example of the production planning problem with $m = 2$ raw materials and $n = 30$ finished products with the following data:

- Purchasing and inventory unit costs of raw materials: $c_i = 100$, $d_i = 150$, and $h_i = 20$ for $i = 1, 2$;

- Production and inventory unit costs of products: $f_j = 530$, $g_j = 750$, and $k_j = 50$ for $j = 1, \ldots, 30$;

- Components of productivity matrix $\mathbf{A}$ were independently drawn from the integer-valued uniform distribution $U[0, 15]$;

- Product demand ranges: $\bar{b}_j = 10$ and $\widehat{b}_j = 5$ for $j = 1, \ldots, 30$.

In our example, $\mathbf{g} > \mathbf{A}^T\mathbf{h} + \mathbf{k}$, ensuring that any production $\mathbf{v}$ in the second stage is performed solely in order to satisfy demand not filled by first-stage production $\mathbf{u}$ (see problem (4.9)).

The green curve in Figure 4.1 (and in Figure 4.2) shows the worst-case cost of this problem as a function of the budget of uncertainty $\Gamma \in [0, 30]$. To assess the expected performance of robust solutions, we considered two possible distributions of demand: one in which demands for individual products follow independent normal distributions with mean $\bar{b}_i = 10$ and standard deviation $\widehat{b}_i = 5$ (truncated at zero to avoid negative values) and one in which demands follow independent continuous uniform distributions on the interval $[\bar{b}_i - \widehat{b}_i, \bar{b}_i + \widehat{b}_i] = [5, 15]$. We generated independent samples of 5000 realizations of the demands for each distribution and plotted the resulting average cost in Figures 4.1 and 4.2 (the red curves; the blue error bars reflect the sample standard deviations) for the normal and uniform distributions, respectively. Just as in the newsvendor problem of Section 3.2, the average cost first decreases with Γ, as incorporating uncertainty into the model yields more robust solutions, reaches its minimum, and then increases with Γ as the solutions become overly conservative. The minimum average cost in Figure 4.1 occurs at $\Gamma = 9$, and by implementing the corresponding ordering and production planning solution, the decision-maker can achieve average savings of 3.3% over the solution obtained for $\Gamma = 0$ (i.e., the solution targeted to satisfy the nominal demand $\overline{\mathbf{b}}$); in Figure 4.2 the minimum occurs at $\Gamma = 6$ with average savings of 2.5%.

Additionally, the 99.96% confidence interval surrounding the mean of the differences (for each demand instance we subtracted the minimum Γ cost from the $\Gamma = 0$ cost and took the average of these differences) is far from containing zero for both the normal and uniform demand sample (see Table 4.1 for the statistical information and confidence interval for each demand distribution). Therefore, we can say with high statistical significance, $p = 0.0004$, that the average cost at $\Gamma = 9$ for the normal demand sample and $\Gamma = 6$ for the uniform demand sample is lower than $\Gamma = 0$, resulting in a savings in average cost.

Demand Distribution	Mean of Differences	Std Dev of Differences	99.96% Confidence Interval
Normal	24783.39	8451.31	$[24366.27, 25200.51]$
Uniform	17435.96	8058.29	$[17038.23, 17833.68]$

Table 4.1: 99.96% confidence intervals for the mean of the differences for the normal and uniform demand samples.

These outcomes make intuitive sense, since demands in the sample generated from the normal distribution exhibit higher variability than in the sample generated from the uniform distribution. This example illustrates the advantage of the robust optimization approach in situations when precise estimates of probability distributions of uncertain parameters are unavailable or inaccurate. Observe that implementing any of the robust solutions obtained by setting Γ anywhere in the range between five and ten yields a robust solution that would perform well (as measured by expected cost) regardless of whether the demands follow a normal or uniform distribution. We performed a number of additional experiments with demands sampled from a variety of distributions. Results presented are typical of all these experiments, with the minimum average cost occurring at $\Gamma \in [5, 15]$, which indicates that in the production planning problem a fair amount of uncertainty needs to be considered to obtain solutions that perform well in expectation.

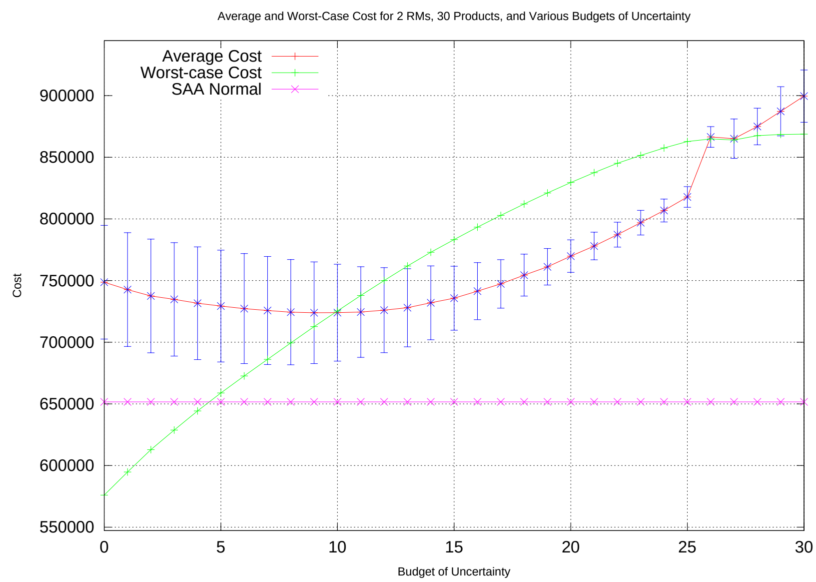


Figure 4.1: The impact of the budget of uncertainty on worst-case and average cost of the production planning problem and the SAA cost under a normal demand distribution.

In an attempt to compare the robust methodology to stochastic programming, we solved the sample average approximation (SAA) of the stochastic version of the production planning problem for the normal and uniform demand samples that were used to assess the average performance of the robust solutions. Thus, the magenta lines in Figures 4.1 and 4.2 represent the cost of the sample average approximation ordering and production policy against the normal and uniform demand samples, respectively. As expected, the sample average approximation cost is lower for both demand distributions; however, the minimum average costs resulting from the robust

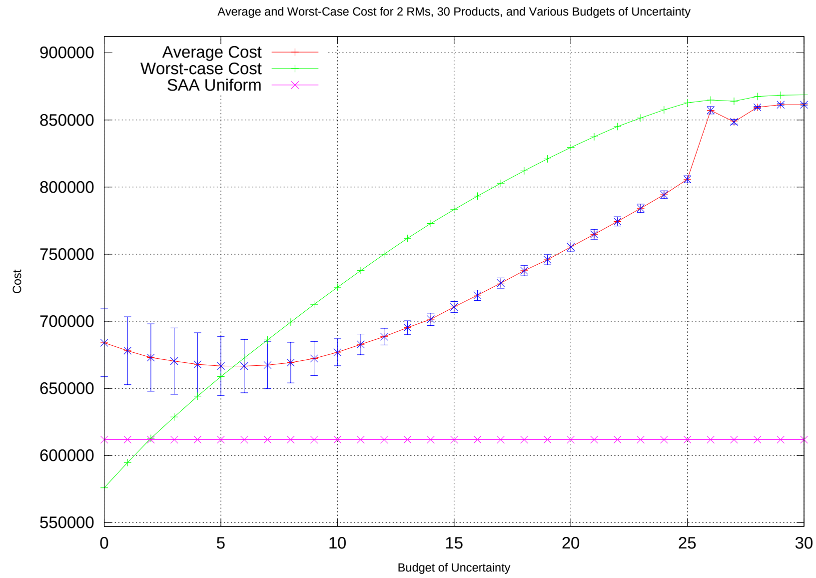


Figure 4.2: The impact of the budget of uncertainty on worst-case and average cost of the production planning problem and the SAA cost under a uniform demand distribution.

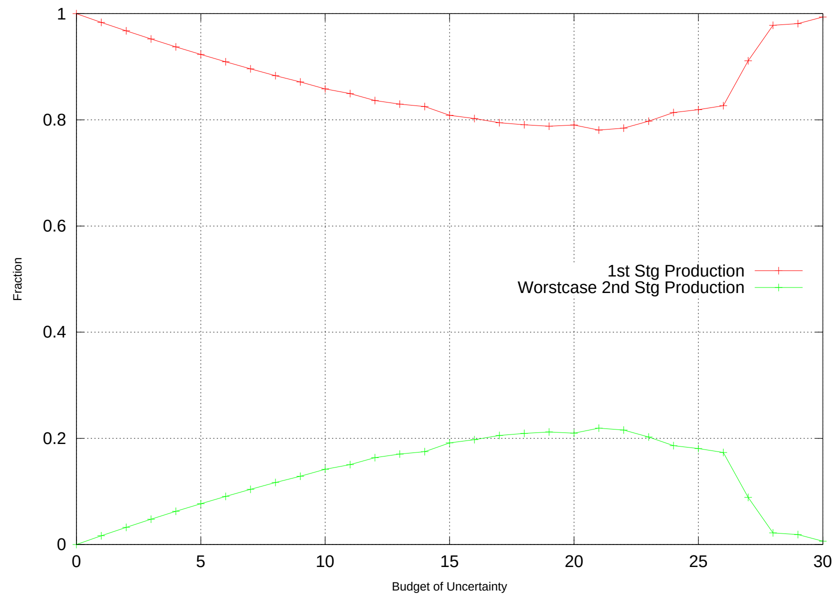
methodology are only 11% and 8.9% larger than the sample average approximation cost for the normal and uniform demand samples, respectively. If the demands are coming from a distribution known to be normal (uniform) with a known mean and standard deviation, then solving the stochastic approximation would provide a lower costing ordering and production policy, but if the distribution is unknown or uncertain prior to realization of the demands, then the robust methodology would provide a good ordering and production policy at a slightly higher cost.

It is informative to consider the amounts of raw materials purchased and production done in the first and second stages. The first-stage purchasing and production decisions ($\mathbf{x}$ and $\mathbf{u}$, respectively) are made according to the solution calculated by solving the robust problem, problem (4.8). Once the first-stage decisions are made and implemented, the actual demand is revealed and the second-stage decisions ($\mathbf{y}$

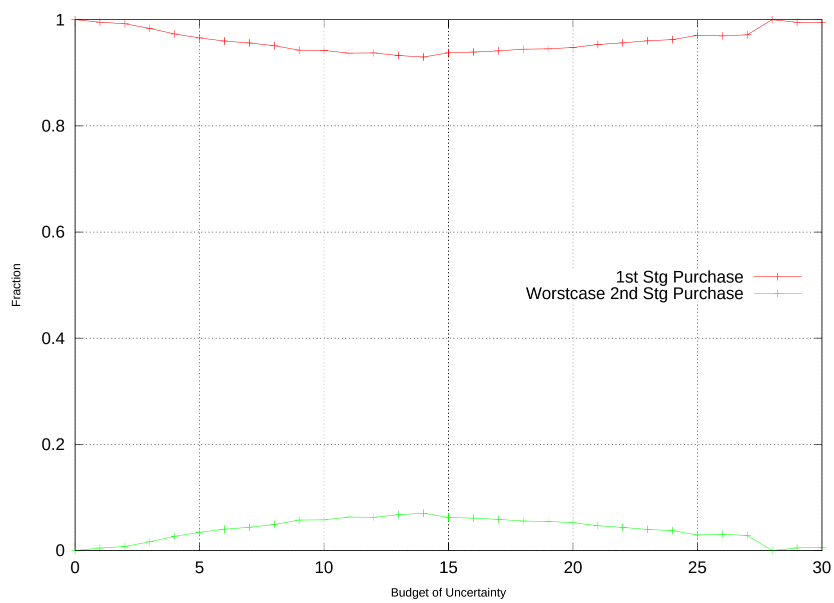
and $\mathbf{v}$) tune themselves to the realized demands.

Figures 4.3(a) and 4.3(b) plot the total amount of raw materials purchased and products produced, respectively, in the first and second stage as fractions of the total purchasing/production that occurs under the worst-case demand outcome. For $\Gamma = 0$, we start with first-stage purchasing/production levels targeted to satisfy nominal demand. As Γ increases, we split purchasing and production between the two stages. However, for higher values of Γ worst-case demand realizations tend to have values higher than nominal, and thus we see the second-stage purchasing and production decreasing.

Figure 4.4(a) displays the sample averages of the fractions of total amounts of raw materials purchased in the first and second stage when first-stage decisions are obtained by solving the robust formulation for various values of Γ and the demands are normally distributed. (Figure 4.4(b) captures similar information for total production, and Figures 4.5(a) and 4.5(b) display these fractions for uniformly distributed demands.) The errors bars show the standard deviation of these average fractions. With uniformly distributed demands, first-stage decisions obtained using high values of Γ in the description of the uncertainty set actually satisfy the realized demand in most cases, as the average second-stage purchasing is zero, and second-stage production is nearly zero, as Figures 4.5(a) and 4.5(b) show (recall also that for these high values of Γ, the average cost of these solutions is almost equal to their worst-case costs). With normally distributed demands, the average fraction of raw material purchases done in the second stage is nearly zero for $\Gamma \in [20, 25]$, but increases for solutions obtained with higher values of Γ; the average fraction of second-stage production decreases with Γ, but never reaches zero. Again, this behavior can partially be explained by higher variability of demands under normal distribution.

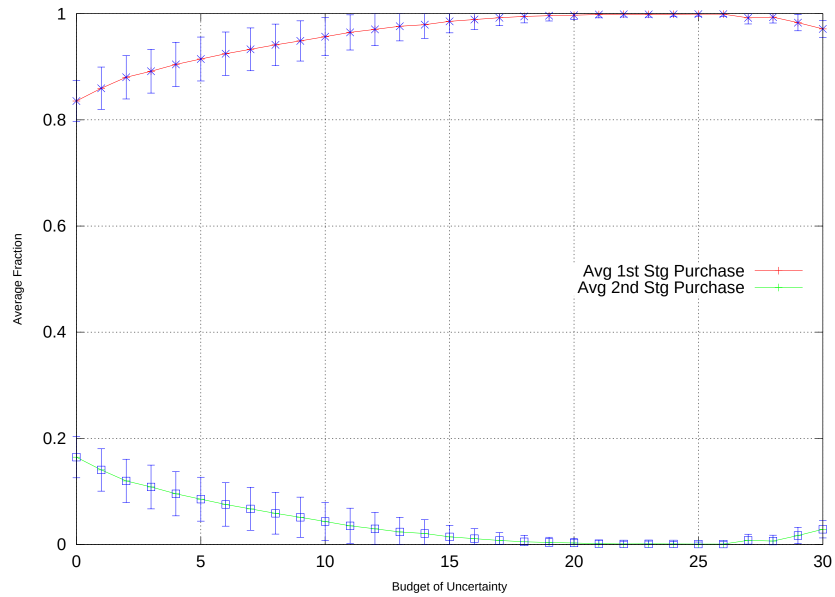


(a) Raw materials purchased.

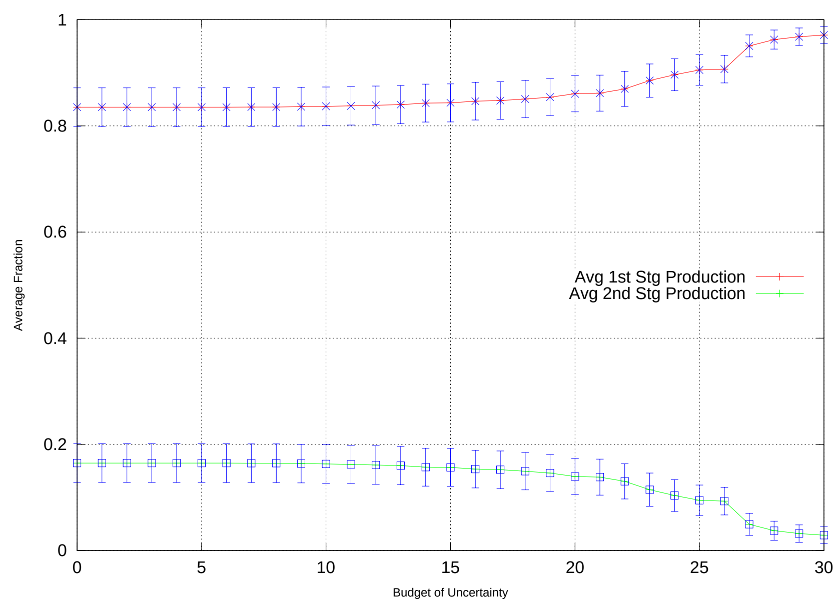


(b) Production performed.

Figure 4.3: First- and second-stage purchasing and production, as fractions of the total purchasing and production, under worst-case demand outcome.

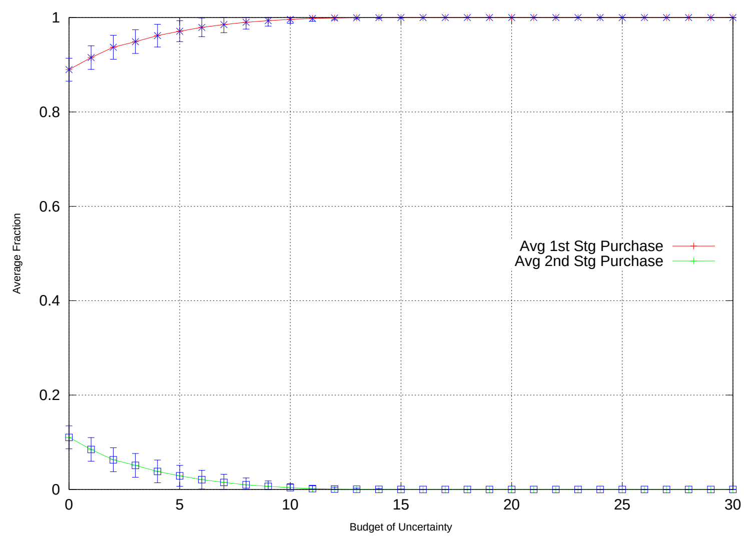


(a) Raw materials purchased, normally distributed demands.

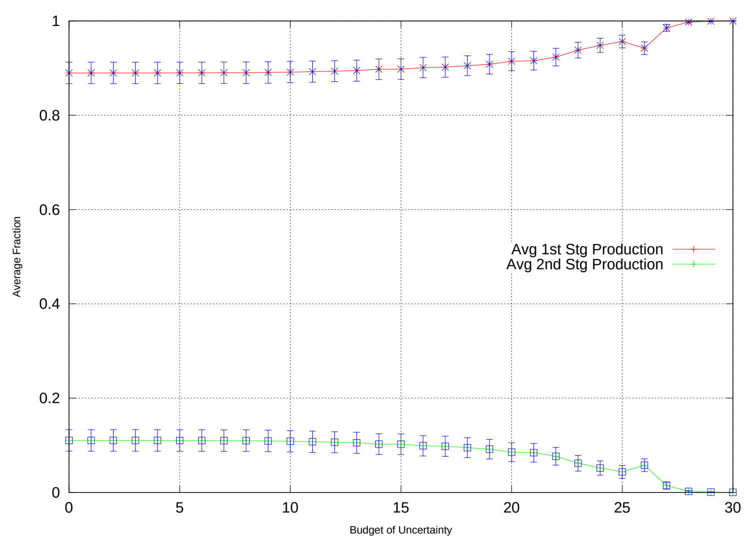


(b) Production performed, normally distributed demands.

Figure 4.4: Sample averages of first- and second-stage purchasing and production, as fractions of the total purchasing and production, under a normal demand distribution.



(a) Raw materials purchased, uniformly distributed demands.



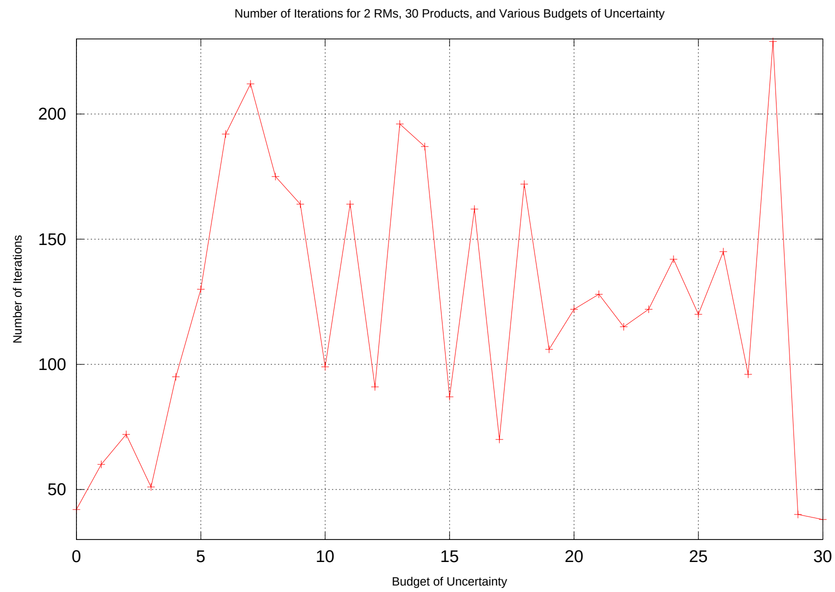
(b) Production performed, uniformly distributed demands.

Figure 4.5: Sample averages of first- and second-stage purchasing and production, as fractions of the total purchasing and production, under a uniform demand distribution.

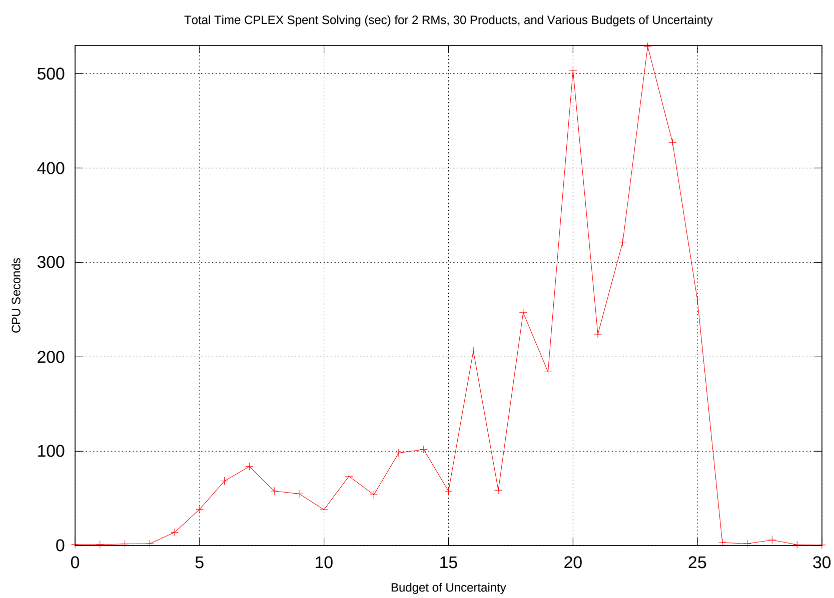
Figures 4.6(a) and 4.6(b) summarize our computational experience by illustrating the effect of the budget of uncertainty on the number of iterations and the running time of Algorithm 2.20 on this problem. The number of iterations required is not particularly sensitive to the value of Γ (except for very small and very large values of Γ) and therefore the determining factor in the running time is the computational demand of solving the mixed-integer program (4.14), which generally increases with Γ. Therefore, the overall running time of the algorithm generally increases with Γ, up to $\Gamma = 22$, and then drops off sharply. Thus, the algorithm has fairly low computational demands for budgets of uncertainty of $\Gamma = 15$ and lower, which were most appropriate for determining robust solutions with low average costs in this and other experiments.

It should be pointed out that, in addition to the value of Γ, the computational demands of the adversarial problem were greatly influenced by the density of the productivity matrix $\mathbf{A}$. Indeed, if the productivity matrix is dense (as it is in the example presented here), all the raw materials would contribute roughly equally towards the production of most or all of the products, which makes it harder to determine which products (and implicitly which raw materials) are more sensitive to demand fluctuations. In examples where a sparse productivity matrix with pronounced block structure allowed the decisions to be implicitly decomposed by materials, instances of the mixed-integer program (4.14) were easier and required shorter solution times.

Finally, we would like to remark that, depending on the relative magnitudes of problem parameters (e.g., the relative magnitudes of inventory and first- and second-stage production costs), one can devise heuristics that would generate an approximate solution to the adversarial problem (4.11) (i.e., a "bad," if not worst, demand realization) quite easily – almost as easily as solving the adversarial problem to optimality in the case of simple recourse. If the demand realization found is worse, in the adversarial sense, than realizations already considered, it can be used to produce a weak cut that separates the current iterate from the epigraph of $Q(\mathbf{x}, \mathbf{u})$, but is typically not a supporting hyperplane (thus, increasing the number of iterations, but possibly leading to faster overall solution times). Alternatively, an algorithm solving (4.14) can be terminated prematurely once a high-quality incumbent solution has been found. In fact, in our experiments we observed that the optimal solution of (4.14) was discovered relatively quickly, and, as is often the case, the bulk of the solution time was spent improving dual bounds and proving optimality of the incumbent solution.



(a) Iteration count vs. Γ.



(b) Run time (CPU seconds) vs. Γ.

Figure 4.6: The impact of the budget of uncertainty on the number of iterations and run time (CPU seconds) for production planning problem.

CHAPTER V

Bounds on Distance to Ill-posedness for Robust Linear Optimization Problems

This chapter investigates the relationship between the conditioning of a robust linear feasibility problem with ellipsoidal uncertainty under structured transformations and the conditioning of the equivalent second-order cone feasibility problem (SOCP) under structured perturbations. Starting from the geometry of the ellipsoidal uncertainty set in data space, we consider various and meaningful structured changes to the data of the robust counterpart and examine the effects on the equivalent SOCP in terms of the standard structured additive perturbation for which the distance to ill-posedness has been previously studied. Our examination results in upper bounds on the distance to ill-posedness of the SOCP in terms of the data given in the initial description of the ellipsoid. In this chapter, we provide a brief description of condition number theory literature, discuss how our work relates to the condition number literature, propose structured changes to the ellipsoidal uncertainty set data, and lastly present our results that bound the distance to ill-posedness of the SOCP by quantities involving the data defining the original ellipsoid.

5.1 Literature Review

The concepts of distance to ill-posedness and condition numbers for optimization problems were introduced by Renegar in [57], who showed that a linear program has large optimal solutions (a sensitive optimal objective value) only if the primal data instance is nearly infeasible or the dual data instance is nearly infeasible. Distance to ill-posedness is an intuitive concept: the amount by which a data instance needs to be perturbed to obtain an infeasible primal or dual data instance. The condition number is a scale-invariant inverse of the distance to ill-posedness, and it reflects the difficulty of the problem instance to be solved. Renegar generalized these concepts more fully in [58] and [59] to convex optimization and feasibility problems in conic linear form and developed complexity theory that allowed for the analysis of iterative algorithms (such as interior-point methods and the ellipsoid algorithm) in terms of the problem-instance size, which is a direct generalization of the condition number and distance to ill-posedness. In addition to Renegar's work, further analysis of condition numbers and their role in the theoretical complexity of solving conic convex optimization problems and convex feasibility problems in conic linear form has been done by Cheung and Cucker [23, 24, 25], Epelman and Freund [30, 31], Filipowski [32, 33], Freund and Vera [36, 37, 38], Nunez and Freund [49], Peña [54], Peña and Renegar [52], Renegar [58, 60], Vera [64, 65], and references therein. In [34], Freund and Ordóñez show that much of the conic-based condition number theory can be naturally extended to non-conic convex optimization problems where the variable is restricted to a convex ground-set.

In addition to theoretical research, there has been some computational work on how condition numbers relate to the number of iterations performed by interior-point methods in practice by Freund, Ordóñez, and Toh. In [50], Ordóñez and Freund show that for the NETLIB suite of linear optimization problems, a statistically significant positive linear relationship exists between the number of interior-point method iterations needed to solve the problems and the log of their condition numbers (after CPLEX pre-processing). In [35], Freund, Ordóñez, and Toh show that two of out the four measures of problem-instance conditioning considered are correlated with the number of interior-point method iterations when solving semi-definite problems from the SDPLIB suite.

Another avenue of research, one tied more closely to the work presented in this chapter, looks at structured perturbations: perturbations restricted to a particular block structure. The distance to infeasibility in the above-mentioned work (with the exception of Filipowski [33]) is derived under the assumption that the relevant problem data is subject to arbitrary and unstructured perturbations. However, data instances can have a pronounced structure such as those resulting from a sparsity pattern, slack variables, or box constraints. In this vein, Peña ([53, 55, 56]) has shown that a natural generalization of Eckart and Young's identity holds for conic systems under block-structured perturbations in various settings.

The robust optimization modeling approach and the study of condition numbers and problem well-posedness are both concerned with subjecting the data of an optimization problem to perturbations. One of the goals of the latter is to provide an assessment of the impact perturbations can have on the optimal solutions and values, whereas the former takes a proactive modeling approach to immunize the solutions of the problem to such perturbations. (For a more detailed comparative discussion, see the following sections.) Due to these similarities, it appeared natural to study the issues of problem conditioning in the context of data perturbations as considered in formulations of robust counterparts.

This chapter presents an initial step in this study. Specifically, we consider a robust counterpart of a linear feasibility problem subjected to data perturbations characterized by an ellipsoidal uncertainty set, and propose several possible measures of distance to ill-posedness for such a problem. We connect these measures to the size and shape of the uncertainty set used to define the robust problem, and provide bounds on these measures in terms of the traditional structured distance to ill-posedness of an equivalent conic feasibility problem, thus grounding our ideas in established concepts.

The eventual goal of this line of research is to develop measures of conditioning for robust optimization problems that would be helpful in understanding the interplay between the choice of the uncertainty set and the impact of that choice on the structure and objective function value of the resulting robust problem, and provide modeling guidance in selecting uncertainty sets that are appropriate for the problem at hand.

5.2 Distance to Infeasibility of a Linear Conic Problem

Consider the general homogenized conic feasibility problem:

$$
\begin{aligned}
&Ax \in C_Y \\
&x \in C_X,
\end{aligned}
\tag{5.1}
$$

where $C_X \subset X$ and $C_Y \subset Y$ are closed convex cones in the finite n-dimensional normed linear vector space X and in the finite m-dimensional normed linear vector space Y, respectively, and $A : X \to Y$ is a linear operator ($A \in \mathbb{R}^{m \times n}$). We will refer to A as a *feasible* data instance when problem (5.1) has a non-trivial (non-zero) solution. If A is changed by some small amount ΔA, would problem (5.1) still be feasible with data instance $A + \Delta A$? How large can ΔA be before problem (5.1) is infeasible? These questions lead toward determining the smallest perturbation ΔA that can be added to A to make $A + \Delta A$ an infeasible data instance, thus arriving at the traditional distance to infeasibility for problem (5.1) introduced by Renegar in [57]:

$$\rho(A) = \inf\{\|\Delta A\| \mid (A + \Delta A)x \in C_Y, x \in C_X \text{ has no non-trivial solutions}\}. \tag{5.2}$$

$\rho(A)$ reflects how problem (5.1) reacts to perturbations or changes in its data A. In [58], Renegar also defined the condition number of the data instance A as

$$\mathcal{C}(A) := \frac{\|A\|}{\rho(A)} \tag{5.3}$$

when $\rho(A) > 0$ and $\mathcal{C}(A) = \infty$ when $\rho(A) = 0$. Problem (5.1) is well-conditioned to the extent that $\mathcal{C}(A)$ is small; when problem (5.1) is ill-posed or ill-conditioned (i.e., arbitrarily small perturbations of the data A can yield both feasible and infeasible problem instances), $\mathcal{C}(A) = +\infty$.

Let us define $\mathcal{F} \subseteq \mathbb{R}^{m\times n}$ to be the set of data instances A for which problem (5.1) has a feasible non-trivial solution, namely

$$\mathcal{F} = \{A \in \mathbb{R}^{m\times n} \mid \exists x \neq 0 \text{ such that } Ax \in C_Y, x \in C_X\}.$$

Then $\mathcal{I} \subseteq \mathbb{R}^{m\times n}$ (the complement of $\mathcal{F}$) consists of data instances A for which problem (5.1) has only a trivial solution. Let the boundary of $\mathcal{F}$ (and of $\mathcal{I}$) be the following set denoted by $\mathcal{B}$:

$$\mathcal{B} = \partial\mathcal{F} = \partial\mathcal{I} = \mathrm{cl}(\mathcal{F}) \cap \mathrm{cl}(\mathcal{I}),$$

where ∂S denotes the boundary set of S and $\mathrm{cl}(S)$ denotes the closure of set S.

Figure 5.1 illustrates an example of the distance to infeasibility for problem (5.1) given feasible data instance A for $\mathcal{F}$, $\mathcal{I}$, and $\mathcal{B}$ as defined above.

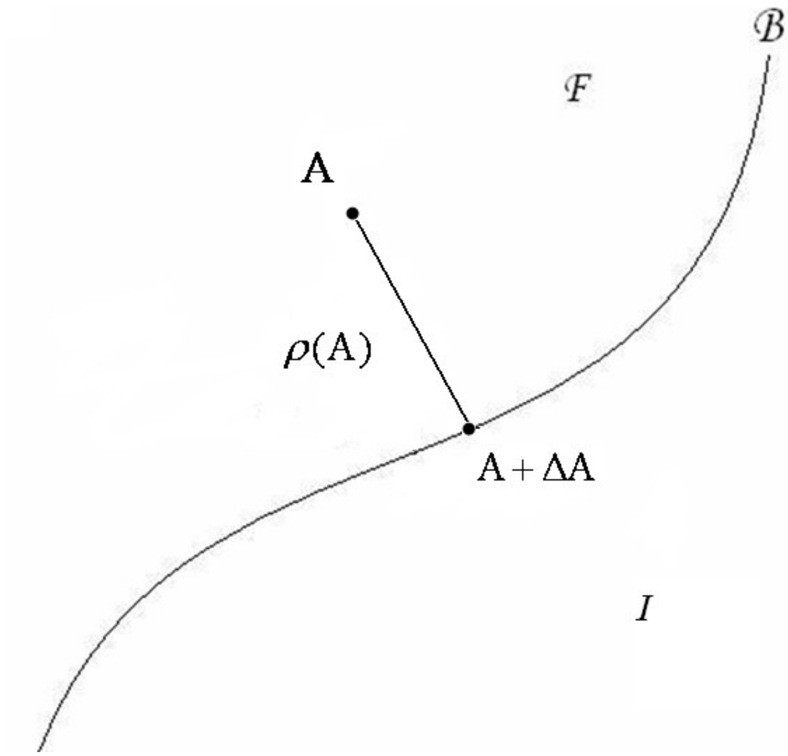


Figure 5.1: Traditional distance to infeasibility for a conic linear feasibility problem.

5.3 Robust Counterpart of LCP

Let LCP be the following problem:

$$
\begin{aligned}
(LCP) \quad & P^0 x \geq 0 \\
& x \in C_X.
\end{aligned} \tag{5.4}
$$

where $C_X \subseteq \mathbb{R}^n$ is a closed convex cone. LCP could be a homogenization of a traditional system of constraints, e.g., if $P^0 = [A \ \ -b]$, $x = \begin{bmatrix} y \\ t \end{bmatrix}$, and $C_X = C \times \mathbb{R}_+$ where C is some closed convex cone, then LCP represents the following linear system:

$Ay \geq bt, y \in C$, and $t \geq 0$.

Notice that LCP is an instance of problem (5.1) with $A = P^0$ and $C_Y = \mathbb{R}^m_+$. Therefore, using the definition given in equation (5.2), the distance to infeasibility for LCP is the following:

$$(5.5)\ \ \rho(P^0) = \inf\{\|\Delta P^0\| \mid (P^0 + \Delta P^0)x \geq 0, x \in C_X \text{ has no non-trivial solutions}\}.$$

Similarly, we can define $\mathcal{F} \subseteq \mathbb{R}^{m\times n}$ to be the set of data instances P^0 for which problem (5.4) has a feasible non-trivial solution,

$$\mathcal{F} = \{P^0 \in \mathbb{R}^{m\times n} \mid \exists x \neq 0 \text{ such that } P^0 x \geq 0, x \in C_X\},$$

$\mathcal{I} \subseteq \mathbb{R}^{m\times n}$ is the set of data instances P^0 for which problem (5.4) has only a trivial solution, and $\mathcal{B}$ is the boundary of $\mathcal{F}$ and of $\mathcal{I}$.

Consider LCP in the robust feasibility framework which mandates that for any perturbation ΔP^0 made to P^0, $P^0 + \Delta P^0$ must be contained in a known and bounded uncertainty set, thereby restricting perturbations to a particular structure and limiting the size of the perturbations. This chapter focuses on ellipsoidal uncertainty sets as considered by Ben-Tal and Nemirovski in [9]. The definition presented by Ben-Tal and Nemirovski is general enough to encompass three types of ellipsoidal uncertainty sets: the standard K-dimensional ellipsoid in $\mathbb{R}^{m\times n}$, flat ellipsoids, and ellipsoidal cylinders. Flat ellipsoids occur when there is partial uncertainty of the data matrix (some data elements are known). Ellipsoidal cylinders occur when there are many ellipsoidal restrictions placed on the matrix P, such as those resulting from having upper and lower bounds on entries of the matrix. As previously mentioned, ellipsoidal uncertainty sets have a nice analytical structure that can be exploited; however, there are modeling advantages for ellipsoidal uncertainty as well: ellipsoidal uncertainty sets cover a wide range of sets (including polytopes by intersecting finitely many ellipsoidal cylinders), ellipsoidal uncertainty sets can be used to approximate complicated convex sets, and an ellipsoidal uncertainty set could be constructed as the minimum-volume ellipsoid containing several given scenarios. Additionally, if the data is of a statistical nature, mutually independent and symmetrically distributed within some interval around a known mean value, then the data uncertainty can be formulated as an ellipsoidal uncertainty set (see Ben-Tal and Nemirovski [9] for a simple portfolio problem with known nominal returns and known bounds on the returns).

The robust counterpart of LCP under ellipsoidal uncertainty, as presented by Ben-Tal and Nemirovski in [9] is:

$$\begin{aligned} &Px \geq 0 \quad \forall P \in \mathcal{E} \\ &x \in C_X, \end{aligned}$$

where $P^0, P^1, \ldots, P^K \in \mathbb{R}^{m\times n}$ are given matrices and

$$\mathcal{E} = \{P = P^0 + \sum_{j=1}^{K} u_j P^j \mid \|u\|_2 \leq 1\} \subset \mathbb{R}^{m\times n}. \tag{5.6}$$

Given the various ways the ellipsoidal uncertainty set can be constructed, the P^j's defining $\mathcal{E}$ might not have any nice properties such as spanning $\mathbb{R}^{m\times n}$, being linearly independent, being orthogonal to one another, or being the axes of the ellipsoid, and therefore no assumptions are made on the P^j's unless otherwise stated.

Substituting the definition of $\mathcal{E}$ produces the following robust counterpart (RC) that will be used throughout the remainder of this chapter:

$$\begin{aligned} (RC): \quad & P^0 x + \sum_{j=1}^{K} u_j P^j x \geq 0 \quad \forall u : \|u\|_2 \leq 1 \\ & x \in C_X. \end{aligned} \tag{5.7}$$

Let P_i^j denote the i^{th} row of matrix P^j and

$$R_i^T = \begin{bmatrix} P_i^1 \\ \vdots \\ P_i^K \end{bmatrix} \in \mathbb{R}^{K \times n}, \quad \tilde{R}_i^T = \begin{bmatrix} P_i^0 \\ R_i^T \end{bmatrix} \in \mathbb{R}^{(K+1) \times n} \tag{5.8}$$

and

$$\tilde{R}^T = \begin{bmatrix} \tilde{R}_1^T \\ \vdots \\ \tilde{R}_m^T \end{bmatrix} \in \mathbb{R}^{m(K+1) \times n}. \tag{5.9}$$

The RC (5.7) is a semi-infinite optimization problem, but it can be transformed into an equivalent second-order cone problem (SOCP) as follows. Note that RC (5.7) is equivalent to

$$\left(\min_{u:\|u\|_2 \leq 1} P_i^0 x + u^T R_i^T x \right) \geq 0 \quad \forall i = 1, \ldots, m.$$

Applying standard first-order optimality conditions to the problem of minimizing $P_i^0 x + u^T R_i^T x$ with respect to u subject to the constraint $\|u\|_2 \leq 1$, we then obtain the following equivalent SOCP (Ben-Tal and Nemirovski [9]):

$$\begin{aligned} (SOCP): \quad & P_i^0 x - \|R_i^T x\|_2 \geq 0 \quad \forall i = 1, \ldots, m \\ & x \in C_X, \end{aligned} \tag{5.10}$$

which, written in the following conic form using definitions (5.8) and (5.9):

$$\begin{aligned} & \tilde{R}^T x \in C_Y \\ & x \in C_X, \end{aligned} \tag{5.11}$$

is another instance of problem (5.1) with $A = \tilde{R}$ and $C_Y = C_{Y_1} \times \ldots C_{Y_m} \subseteq \mathbb{R}^{m(K+1)}$ where, for $i = 1, \ldots, m$, $C_{Y_i} \subseteq \mathbb{R}^{K+1}$ is a second-order cone.

In this chapter, we will use the following slight abuse of notation to refer to the data specifying equivalent instances of problems (5.7) and (5.10):

- when discussing the RC (problem (5.7)), d refers to the collection of matrices $(P^0, P^1, \ldots, P^K)$ and

- when discussing the SOCP (problem (5.10)), d refers to the augmented matrix

$$d = \tilde{R}^T = \begin{bmatrix} \tilde{R}_1^T \\ \vdots \\ \tilde{R}_m^T \end{bmatrix} \in \mathbb{R}^{m(K+1)\times n},$$

  where $\tilde{R}_i^T$ is defined in equation (5.8).

Similar to the definitions of section 5.2, define $\mathcal{F}_{\text{SOCP}}$ to be the set of data instances d for which the SOCP has a feasible non-trivial solution, namely

$$\mathcal{F}_{\text{SOCP}} = \{d \in \mathbb{R}^{m(K+1)\times n} \mid \exists x \neq 0 \text{ satisfying problem (5.11)}\}.$$

Let $\mathcal{I}_{\text{SOCP}}$ (the complement of $\mathcal{F}_{\text{SOCP}}$) be the set of data instances d for which the SOCP only has a trivial solution. Since $d \in \mathbb{R}^{m(K+1)\times n}$, $\mathcal{F}_{\text{SOCP}} \subset \mathbb{R}^{m(K+1)\times n}$ and $\mathcal{I}_{\text{SOCP}} \subset \mathbb{R}^{m(K+1)\times n}$. Lastly, $\mathcal{B}_{\text{SOCP}}$ is the boundary of $\mathcal{F}_{\text{SOCP}}$ and $\mathcal{I}_{\text{SOCP}}$.

In the remainder of this chapter we proceed to describe four types of structured transformations, indexed by 0, 1, 2, and 3, which change the RC data $d = (P^0, P^1, \ldots, P^K)$ in some manner. The transformed RC data results in perturbed SOCP data $\bar{d} = d + \Delta d \in \mathbb{R}^{m(K+1)\times n}$, which fall into one of two types of structured perturbations. The latter two types of perturbations are detailed in section 5.4.

Mathematical preliminaries

We start by presenting some mathematical definitions and relations that will be referenced throughout the remainder of this chapter, which can be found in Meyer [44] and other standard linear algebra references.

For $p \geq 1$, the p-norm of $x \in \mathbb{R}^n$ is defined as

$$\|x\|_p = \Big(\sum_{i=1}^{n} |x_i|^p\Big)^{\frac{1}{p}}. \tag{5.12}$$

Each pair of vector norms, $\|\cdot\|_a$ and $\|\cdot\|_b$, on an n-dimensional space are equivalent, i.e., there exist positive constants β and γ (dependent upon the choice of norms) such that for all x,

$$\beta\|x\|_a \leq \|x\|_b \leq \gamma\|x\|_a.$$

The vector p-norm defined on $\mathbb{R}^n$ and $\mathbb{R}^m$ induces a matrix norm on $\mathbb{R}^{m\times n}$ by setting:

$$\|A\|_p = \max_{\|x\|_p=1} \|Ax\|_p \tag{5.13}$$

for $A \in \mathbb{R}^{m\times n}$ and $x \in \mathbb{R}^n$. A matrix norm defined on $\mathbb{R}^{m\times n}$ is also a vector norm defined on $\mathbb{R}^{mn}$, thus equivalency holds for pairs of matrix norms as well.

Additionally, given matrices $A \in \mathbb{R}^{m\times m}$ and $B \in \mathbb{R}^{m\times n}$, the p-norm also satisfies the submultiplicative property:

$$\|AB\|_p \leq \|A\|_p\|B\|_p. \tag{5.14}$$

Lastly, we present the following result to be used later in the chapter: given $\alpha \in \mathbb{R}^K$, let $\alpha_{\max}$ denote the maximum element of α, and $\mathrm{diag}(\alpha)$ denote a diagonal matrix with elements of α arranged on the diagonal. Then

$$\begin{aligned} \|\mathrm{diag}(\alpha)\|_p &= \max_{\|x\|_p=1} \|\mathrm{diag}(\alpha)x\|_p \\ &= \max_{\|x\|_p=1} \|\alpha^T x\|_p \\ &= \max_{\|x\|_p=1} \Big(\sum_{i=1}^{K} |\alpha_i x_i|^p\Big)^{\frac{1}{p}} \\ &= \Big((\alpha_{\max}\cdot 1)^p\Big)^{\frac{1}{p}} \\ &= \alpha_{\max} \end{aligned} \tag{5.15}$$

5.4 Structure of perturbations to SOCP

This section discusses a paper from Ben-Tal and Nemirovski that highlights the need to consider data uncertainty, as well as work from Peña, [55] in particular, who considers block-structured perturbations that are similar in nature to those being considered in this chapter. We then present two types of structured perturbations for the SOCP and their corresponding definitions for distance to infeasibility under each type of structured perturbation, which will be referenced throughout the remainder of this chapter.

In [10], Ben-Tal and Nemirovski studied the optimal solutions of 90 linear programs from the NETLIB library to: quantify the level of infeasibility of the nominal solution in the face of small uncertainty, apply the robust optimization methodology when the level of infeasibility is large to obtain a solution that is immunized against data perturbations, and determine the price of robustness. However, theirs considered partial data uncertainty: they coined the term "ugly reals" (e.g. 15.79081 or 84.644257) that may characterize certain technological devices or processes and usually are known to within three or four digits of accuracy, but no more. Therefore, these "ugly reals" are uncertain. On the other hand, coefficients of zeros and ones seem to reflect the structure of the problem and are consequently certain. They show for one constraint from the PILOT4 LP that when the uncertain data is perturbed by 0.1%, the worst-case violation can be as large as 450% of the right-hand side, while the typical violation when perturbations are independently and uniformly distributed between [-0.1%, 0.1%] results in an average violation of 125% of the right-hand side. Thus, the work presented in this paper is important for two reasons: it demonstrates the need to consider robust solutions when the problem data is subject to perturbations, and secondly, gives real examples of structured data perturbations, as reflected in "certain" and "uncertain" coefficients.

With this distinction of certain and uncertain coefficients, Peña's work in [55] considers block-structured perturbation where only some of the coefficients are subject to perturbations. Consider the homogenized linear system (and subsequent definitions) as presented by Peña in [55]:

$$Ax = 0, x \in C$$

where $A \in \mathbb{R}^{m\times n}$ and C is a closed convex cone. Peña considers several types of block-structured perturbations: low-rank perturbations, horizontal block-structured perturbations, and general block-structured perturbations. As an example, the horizontal block-structured perturbation occurs when the subspaces X_i, where $\mathbb{R}^n = X_1 \times \ldots \times X_k$, form a direct decomposition of $\mathbb{R}^n$ and $Y_1, \ldots, Y_k$ are linear subspaces of $\mathbb{R}^m$. The horizontal block-structured perturbation is then:

$$\Delta A = \begin{bmatrix} \Delta A_1 & \ldots & \Delta A_k \end{bmatrix} \tag{5.16}$$

where $\Delta A_i : X_i \rightarrow Y_i,\ i = 1, \ldots, k$. Given $\alpha_1, \ldots, \alpha_k \geq 0$, the scaled norm is defined as the following:

$$\|\Delta A\|_\alpha = \max\left\{\frac{\|\Delta A_i\|}{\alpha_i} \mid \alpha_i \neq 0,\ i = 1, \ldots, k\right\}$$

The distance to infeasibility, denoted $dist_{blk}(A)$, is then:

$$dist_{blk}(A) = \inf\left\{\|\Delta A\|_\alpha \mid \Delta A \text{ has form (5.16) s.t. } (A+\Delta A)x = 0, x \in C \text{ is inconsistent}\right\}.$$

Our work is similar in that we both consider block-structured perturbations that perturb only some of the data in a particular fashion, but the main difference in [55] is Peña's key proof technique underlying his results: a low-rank construction of the minimal infeasible perturbation, an extension of his rank-one approach used in

[53], which he uses to extend Eckart and Young's identity for distance to infeasibility of conic systems under block-structured perturbations. We do not attempt to characterize the minimal infeasible perturbation, but rather study the relationship between various distances to infeasibility of a robust counterpart under particular data transformations and the distance to infeasibility of the equivalent conic system under structured perturbations.

The two types of structured perturbations to the SOCP data we are considering deal with perturbing particular rows of d. The first type of structured perturbation $\Delta_I d$ only perturbs the first row of each $\tilde{R}_i^T$ block:

$$\Delta_I d = \begin{bmatrix} \begin{bmatrix} \Delta P_1^0 \\ 0 \end{bmatrix} \\ \vdots \\ \begin{bmatrix} \Delta P_m^0 \\ 0 \end{bmatrix} \end{bmatrix}. \tag{5.17}$$

Definition 5.18. Let the SOCP distance to infeasibility under perturbation structure I be:

$$\rho_I(d) = \inf_{d+\Delta_I d \in \mathcal{I}_{\text{SOCP}}} \{\|\Delta_I d\|_p \mid \Delta_I d \text{ has structure (5.17)}\}. \tag{5.19}$$

where $\|\Delta_I d\|_p$ is the matrix p-norm defined in equation (5.13).

Notice that, since the p-norm of a matrix is invariant to deletion of rows consisting entirely of zeros, we can simplify $\|\Delta_I d\|_p$ as shown below:

$$\begin{aligned} \|\Delta_I d\|_p &= \max_{\|x\|_p=1} \|\Delta_I d x\|_p \\ &= \max_{\|x\|_p=1} \|\Delta P^0 x\|_p \\ &= \|\Delta P^0\|_p. \end{aligned} \tag{5.20}$$

The second type of structured perturbation $\Delta_{II}d$ perturbs every row except the first row of each $\tilde{R}_i^T$ block:

$$\Delta_{II}d = \begin{bmatrix} \begin{bmatrix} 0 \\ \Delta R_1^T \end{bmatrix} \\ \vdots \\ \begin{bmatrix} 0 \\ \Delta R_m^T \end{bmatrix} \end{bmatrix}. \tag{5.21}$$

Definition 5.22. Let the SOCP distance to infeasibility under perturbation structure II be:

$$\rho_{II}(d) = \inf_{d+\Delta_{II}d \in \mathcal{I}_{\text{SOCP}}} \{\|\Delta_{II}d\|_p \mid \Delta_{II}d \text{ has structure (5.21)}\}. \tag{5.23}$$

where $\|\Delta_{II}d\|_p$ is the matrix p-norm defined in equation (5.13).

Let R^T be the matrix produced by stacking each R_i^T matrix on top of each other (recall that the R_i^T matrices are produced by stacking the i^{th} row of each P^j matrix $j = 1, \ldots, K$ on top of each other):

$$R^T = \begin{bmatrix} R_1^T \\ \vdots \\ R_m^T \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} P_1^1 \\ \vdots \\ P_1^K \end{bmatrix} \\ \vdots \\ \begin{bmatrix} P_m^1 \\ \vdots \\ P_m^K \end{bmatrix} \end{bmatrix}, \tag{5.24}$$

and let ΔR^T be defined analogously. Again, removal of rows consisting entirely of zeros allows for simplification of $\|\Delta_{II}d\|_p$. Moreover, the p-norm of a matrix is also invariant to permutations of rows of the matrix, and since R^T is a result of row permutations of S, where

$$S = \begin{bmatrix} P^1 \\ \vdots \\ P^K \end{bmatrix}, \tag{5.25}$$

and, analogously, ΔR^T is a result of row permutations of ΔS, we obtain the following simplification of $\|\Delta_{II} d\|_p$:

$$\begin{aligned} \|\Delta_{II} d\|_p &= \max_{\|x\|_p=1} \|\Delta_{II} dx\|_p \\ &= \max_{\|x\|_p=1} \|\Delta R^T x\|_p \\ &= \|\Delta R^T\|_p \\ &= \|\Delta S\|_p. \end{aligned} \tag{5.26}$$

As discussed earlier, the traditional distance to infeasibility gives an indication of how far a data instance A is from the boundary of infeasibility $\mathcal{B}$ and therefore how well-posed (or ill-posed) the convex feasibility problem (5.1) is. We now consider the feasibility status of robust counterparts and well-posedness (or ill-posedness) of the robust counterpart, i.e., how far the ellipsoidal uncertainty set in data space is from the boundary of infeasibility. As $\rho(A)$ measures the size of the perturbation necessary to make problem (5.1) change feasibility status, we propose ways to measure sizes of transformations that make the robust counterpart change feasibility status. With this goal in mind, the remainder of the chapter discusses four types of transformations to the RC data and provides a way to measure the size of the transformation. Additionally, we relate the structured distance to infeasibility of the SOCP to the distance to infeasibility of the RC.

5.5 Shifting the center of the ellipsoid: RC transformation structure 0

Let the ellipsoid defining the original RC be denoted by $\mathcal{E}_d$ where $\mathcal{E}_d$ is robustly feasible ($\mathcal{E}_d \subset \mathcal{F}$) and has center P^0. One way to determine how far $\mathcal{E}_d$ is from the boundary $\mathcal{B}$ is to shift $\mathcal{E}_d$ in data space until $\mathcal{E}_d$ intersects $cl(\mathcal{I})$. This type of transformation, referred to as structure 0, changes the data defining the center of ellipsoid without changing the size or shape of the ellipsoid and is the natural extension of the usual data perturbation for LCP. Moreover, transforming $\mathcal{E}_d$ with center P^0 to $\mathcal{E}_{\bar{d}}$ having center $P^0 + \Delta P^0$ allows us to measure the size of this transformation simply as $\|P^0 + \Delta P^0 - P^0\| = \|\Delta P^0\|$. Figure 5.2 illustrates an example of a structure 0 transformation while Table 5.1 summarizes the changes to the original RC data and ellipsoid under transformation of structure 0.

	Original	Transformed
Data	$d = (P^0, P^1, \ldots, P^K)$	$\bar{d} = (P^0 + \Delta P^0, P^1, \ldots, P^K)$
Ellipsoid	$\mathcal{E}_d = \{P^0 + \sum_{j=1}^{K} u_j P^j \mid \|u\|_2 \leq 1\}$	$\mathcal{E}_{\bar{d}} = \{(P^0 + \Delta P^0) + \sum_{j=1}^{K} u_j P^j \mid \|u\|_2 \leq 1\}$

Table 5.1: Changes to the RC data under structure 0 transformation

This type of transformation seems natural when the nominal data P^0 is uncertain, but the data defining the size and shape of $\mathcal{E}_d$ is known. As an example, consider five machines that each perform a specific type of measurement. Each machine performs and stores this measurement multiple times over a period of time and thus each machine has collected a set of data for which it can determine the mean value and standard deviation. P^0 could represent some kind of aggregate value of the five mean values while P^j could represent the standard deviation of data set j from machine j, $j = 1, \ldots, 5$. The nominal data P^0 might not be a meaningful measure, while

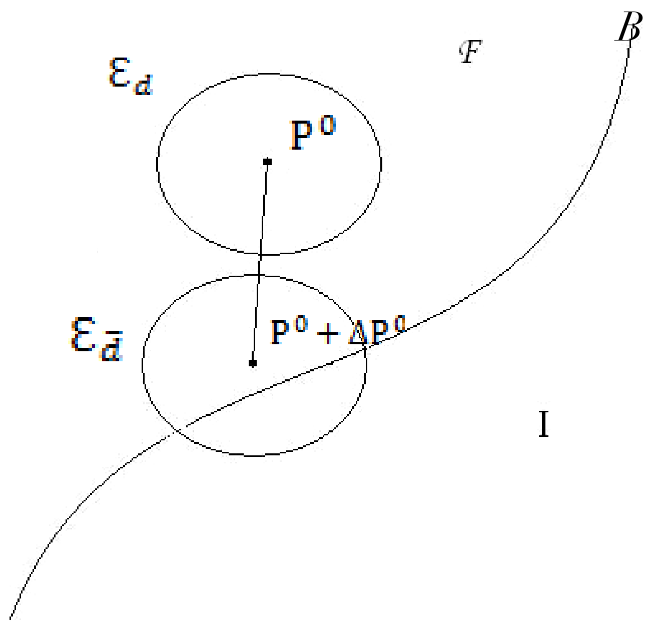


Figure 5.2: Example of RC transformation under structure 0 transformation

$P^1, \ldots, P^5$ are very meaningful. A transformation under structure 0 could indicate the largest error on P^0, while still maintaining robust feasibility.

Definition 5.27. Let the RC distance to infeasibility under transformation structure 0 be:

$$\rho_0(d) = \inf_{\mathcal{E}_{\bar{d}} \cap \mathcal{I} \neq \emptyset} \{ \|\Delta P^0\|_p \mid \bar{d} = (P^0 + \Delta P^0, P^1, \ldots, P^K) \}. \tag{5.28}$$

As is the case with $\rho(P^0)$ defined in equation (5.5), the farther away $\mathcal{E}_d$ is from $\mathcal{I}$, the larger $\rho_0(d)$ becomes.

Rewriting the transformed RC having data $\bar{d} = (P^0 + \Delta P^0, P^1, \ldots, P^K)$ results in the equivalent SOCP with data

$$\bar{d} = \begin{bmatrix} \begin{bmatrix} P_1^0 + \Delta P_1^0 \\ R_1^T \end{bmatrix} \\ \vdots \\ \begin{bmatrix} P_m^0 + \Delta P_m^0 \\ R_m^T \end{bmatrix} \end{bmatrix}.$$

Notice that $\Delta_I d = \bar{d} - d$ corresponds to SOCP perturbation of structure I given in equation (5.17). Table 5.2 summarizes the changes to the SOCP data under perturbation structure I.

	Original	Perturbed
SOCP	$P_i^0 x - \|R_i^T x\|_2 \geq 0 \ \forall i$ $x \in C_X$	$(P_i^0 + \Delta P_i^0)x - \|R_i^T x\|_2 \geq 0 \ \forall i$ $x \in C_X$
Data	$d = \begin{bmatrix} \tilde{R}_i^T \\ \vdots \\ \tilde{R}_i^T \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} P_1^0 \\ R_1^T \end{bmatrix} \\ \vdots \\ \begin{bmatrix} P_m^0 \\ R_m^T \end{bmatrix} \end{bmatrix}$	$\bar{d} = \begin{bmatrix} \begin{bmatrix} P_1^0 + \Delta P_1^0 \\ R_1^T \end{bmatrix} \\ \vdots \\ \begin{bmatrix} P_m^0 + \Delta P_m^0 \\ R_m^T \end{bmatrix} \end{bmatrix}$

Table 5.2: Changes to the SOCP data under structure I perturbation

Proposition 5.29. *Let the norm on the SOCP data $d \in \mathbb{R}^{m(K+1)\times n}$ be the matrix p-norm and the SOCP distance to infeasibility under perturbation structure I be defined as in equation* (5.19). *Let the norm on the RC data be the matrix p-norm and the RC distance to infeasibility under transformation structure 0 be defined as in equation* (5.28). *Then*

$$\rho_I(d) \leq \rho_0(d).$$

Proof. If $\rho_0(d) = +\infty$, then the result is trivially true. Assume $\rho_0(d) < +\infty$. Let $\epsilon > 0$ and let ΔP^0 be such that $\mathcal{E}_{\bar{d}} \cap \mathcal{I} \neq \emptyset$, where $\bar{d} = (P^0 + \Delta P^0, P^1, \ldots, P^K)$ and $\|\Delta P^0\|_p \leq \epsilon + \rho_0(d)$.

Let $\Delta_I d$ be a perturbation of the SOCP data d having structure I obtained by replacing the appropriate rows with rows of ΔP^0, as defined above, in a $m(K+1) \times n$ matrix of zeros. Note that, since $\mathcal{E}_{\bar{d}} \cap \mathcal{I} \neq \emptyset$, $d + \Delta_I d \in \mathcal{I}_{\text{SOCP}}$. Hence,

$$\rho_I(d) \leq \|\Delta_I d\|_p,$$

which results in

$$\rho_I(d) \leq \|\Delta_I d\|_p = \|\Delta P^0\|_p \leq (\epsilon + \rho_0(d)),$$

by equation (5.20). Since the above is true for any $\epsilon > 0$, the desired result follows.

$\square$

Proposition 5.30. *Consider LCP, the original conic system given by problem* (5.1). *Let the distance to infeasibility of LCP be defined by equation* (5.2) *and let the norm on the LCP data be the p-norm. Let the RC distance to infeasibility under transformation structure 0 be defined as in equation* (5.28) *and let the norm on the RC data be the p-norm. Then*

$$\rho_0(d) \leq \rho(P^0).$$

Proof. Let ΔP^0 be such that $P^0 + \Delta P^0 \in \mathcal{I}$. Then the instance of RC with data $\bar{d} = (P^0 + \Delta P^0, P^1, \ldots, P^K)$ is not feasible and thus $\rho_0(d) \leq \|\Delta P^0\|_p$. Therefore,

$$\rho_0(d) \leq \rho(P^0).$$

$\square$

5.6 Transforming the size and shape of the ellipsoid

5.6.1 Common-scaling transformation of the ellipsoid: RC transformation structure 1

Assuming $\mathcal{E}_d$ is robustly feasible ($\mathcal{E}_d \subset \mathcal{F}$) and has center P^0, another way to determine how far $\mathcal{E}_d$ is from the boundary of infeasibility is to inflate, or scale, $\mathcal{E}_d$ in data space until $\mathcal{E}_d$ intersects $cl(\mathcal{I})$. This type of transformation, referred to as structure 1, fixes the center of the ellipsoid and maintains the shape of the ellipsoid, but increases the size of the ellipsoid. The obvious way to measure the size of this perturbation is to look at the magnitude of the size increase. Figure 5.3 illustrates an example of a structure 1 transformation, while Table 5.3 summarizes the changes to the original RC data and ellipsoid under transformation structure 1. The parameter $\alpha > 0$ reflects the magnitude of scaling of the ellipsoid.

	Original	Transformed
Data	$d = (P^0, P^1, \ldots, P^K)$	$\bar{d} = (P^0, \alpha P^1, \ldots, \alpha P^K)$
Ellipsoid	$\mathcal{E}_d = \{P^0 + \sum_{j=1}^{K} u_j P^j \mid \|u\|_2 \leq 1\}$	$\mathcal{E}_{\bar{d}} = \{P^0 + \sum_{j=1}^{K} u_j \alpha P^j \mid \|u\|_2 \leq 1\}$

Table 5.3: Changes to the RC data under structure 1 transformation; here $\alpha > 0$

This type of transformation seems natural when the nominal data P^0 is known, but the data defining the size and shape of $\mathcal{E}_d$ is uncertain. Consider the machine example, but add the modification that the data storage for each machine is getting too costly and management is trying to determine by how much to reduce the data collection and storage for each machine, but now the nominal data is known and certain. The issue with reducing data collection is that the reduced number of samples results in smaller data sets, which cause the standard deviations to increase, resulting in the inflation of the ellipsoid. A transformation under structure 1 would

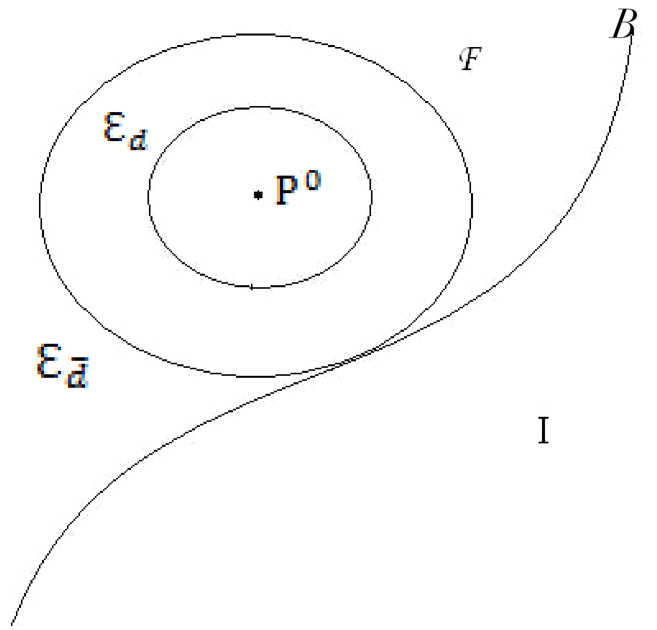


Figure 5.3: Example of RC transformation under structure 1 transformation

indicate in some manner by how much one could reduce the number the samples collected from each machine, but still maintain robust feasibility.

Definition 5.31. Let the RC distance to infeasibility under transformation structure 1 be:

$$\rho_1(d) = \inf_{\mathcal{E}_{\bar{d}} \cap \mathcal{I} \neq \emptyset} \{\alpha \mid \bar{d} = (P^0, \alpha P^1, \ldots, \alpha P^K)\} - 1. \tag{5.32}$$

Since, according to our assumption, $\mathcal{E}_d \subset \mathcal{F}$, one needs to take $\alpha \geq 1$ to obtain an infeasible or nearly infeasible data instance (where $\mathcal{E}_{\bar{d}} \cap \mathcal{I} \neq \emptyset$), which guarantees $\rho_1(d) \geq 0$. The larger $\rho_1(d)$, the farther away $\mathcal{E}_d$ is from $\mathcal{I}$.

Rewriting the transformed RC having data $\bar{d} = (P^0, \alpha P^1, \ldots, \alpha P^K)$ results in the equivalent SOCP with data

$$\bar{d} = \begin{bmatrix} \begin{bmatrix} P_1^0 \\ \alpha R_1^T \end{bmatrix} \\ \vdots \\ \begin{bmatrix} P_m^0 \\ \alpha R_m^T \end{bmatrix} \end{bmatrix}.$$

Notice that $\Delta_{II} d = \bar{d} - d$ corresponds to SOCP perturbation of structure II given in equation (5.21). Table 5.4 summarizes the changes to the SOCP data resulting from this perturbation.

	Original	Perturbed
SOCP	$P_i^0 x - \|R_i^T x\|_2 \geq 0\ \forall i$ $x \in C_X$	$P_i^0 x - \|\alpha R_i^T x\|_2 \geq 0\ \forall i$ $x \in C_X$
Data	$d = \begin{bmatrix} \tilde{R}_i^T \\ \vdots \\ \tilde{R}_i^T \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} P_1^0 \\ R_1^T \end{bmatrix} \\ \vdots \\ \begin{bmatrix} P_m^0 \\ R_m^T \end{bmatrix} \end{bmatrix}$	$\bar{d} = \begin{bmatrix} \begin{bmatrix} P_1^0 \\ \alpha R_1^T \end{bmatrix} \\ \vdots \\ \begin{bmatrix} P_m^0 \\ \alpha R_m^T \end{bmatrix} \end{bmatrix}$

Table 5.4: Changes to the SOCP data under structure II perturbation resulting from a structure 1 transformation on the RC data.

Proposition 5.33. *Let the norm on the SOCP data $d \in \mathbb{R}^{m(K+1)\times n}$ be the matrix p-norm and the SOCP distance to infeasibility under perturbation structure II be defined as in equation* (5.23)*. Let the norm on the RC data be the matrix p-norm and the RC distance to infeasibility under transformation structure 1 be defined as in equation* (5.32)*. Then*

$$\rho_{II}(d) \leq \|S\|_p \rho_1(d).$$

where S, defined in equation (5.25)*, is the matrix produced by stacking the P^j's, $j = 1, \ldots, K$ on top of each other.*

Proof. If $\rho_1(d) = +\infty$, then the result is trivially true. Assume $\rho_1(d) < +\infty$. Let $\epsilon > 0$ and let α be such that $\mathcal{E}_{\bar{d}} \cap \mathcal{I} \neq \emptyset$, where $\bar{d} = (P^0, \alpha P^1, \ldots, \alpha P^K)$ and $\alpha - 1 \leq \epsilon + \rho_1(d)$.

Let $\Delta_{II} d$ be a perturbation of the SOCP data d having structure II obtained by replacing the appropriate blocks of zeros with blocks $\Delta R_i^T = (\alpha - 1) R_i^T$, $i = 1, \ldots, m$ in a $m(K+1) \times n$ matrix of zeros. Note that, since $\mathcal{E}_{\bar{d}} \cap \mathcal{I} \neq \emptyset$, $d + \Delta_{II} d \in \mathcal{I}_{\text{SOCP}}$. Then we have:

$$\begin{aligned}
\|\Delta_{II} d\|_p &= \max_{\|x\|_p = 1} \|\Delta_{II} d x\|_p \\
&= \max_{\|x\|_p = 1} \|(\alpha - 1) R^T x\|_p \\
&= (\alpha - 1) \max_{\|x\|_p = 1} \|R^T x\|_p \\
&= (\alpha - 1) \|R^T\|_p \\
&= (\alpha - 1) \|S\|_p.
\end{aligned}$$

Hence,

$$\rho_{II}(d) \leq \|\Delta_{II} d\|_p,$$

which results in

$$\rho_{II}(d) \leq \|\Delta_{II} d\|_p = (\alpha - 1)\|S\|_p \leq \|S\|_p(\epsilon + \rho_1(d)).$$

Since the above is true for any $\epsilon > 0$, the desired result follows. $\square$

5.6.2 Independent-scaling transformation of the ellipsoid: RC transformation structure 2

Assuming $\mathcal{E}_d$ is robustly feasible ($\mathcal{E}_d \subset \mathcal{F}$) and has center P^0, a third way way to determine how far $\mathcal{E}_d$ is from the boundary of infeasibility is to amplify each P^j

independently until at least one point in $\mathcal{E}_d$ intersects $cl(\mathcal{I})$. This type of transformation, referred to as structure 2, fixes the center of the ellipsoid, but changes the size and shape of the ellipsoid in a particular way. The way we propose to measure this type of transformation is to determine which P^j received the largest scaling factor. If the P^j's were the axes of the ellipsoid, this would tell you which axis received the largest increase in length to reach infeasibility. Figure 5.4 illustrates an example of a structure 2 transformation where the P^j's are the axes of the ellipsoid. As you can see, under a structure 2 transformation, each P^j is scaled independently and so the length of each P^j changes, but not the direction. Table 5.5 summarizes the changes to the original RC data and ellipsoid under transformation structure 2.

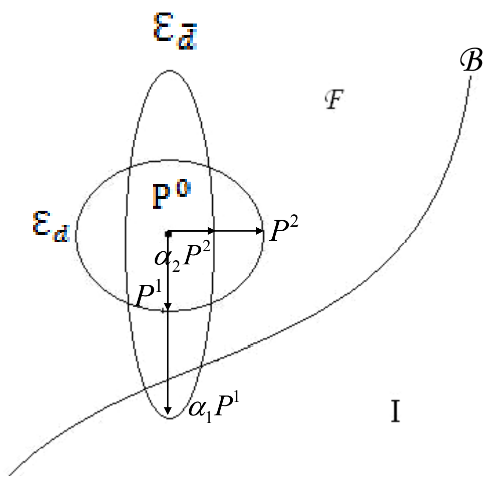


Figure 5.4: Example of RC transformation under structure 2 transformation

	Original	Transformed
Data	$d = (P^0, P^1, \ldots, P^K)$	$\bar{d} = (P^0, \alpha_1 P^1, \ldots, \alpha_K P^K)$
Ellipsoid	$\mathcal{E}_d = \{P^0 + \sum_{j=1}^{K} u_j P^j \mid \|u\|_2 \leq 1\}$	$\mathcal{E}_{\bar{d}} = \{P^0 + \sum_{j=1}^{K} u_j \alpha_j P^j \mid \|u\|_2 \leq 1\}$

Table 5.5: Changes to the RC data under structure 2 transformation

As the structure 2 transformation is similar to the structure 1 transformation, structure 2 transformations would occur when the nominal data P^0 is known, but the data defining the size and shape of $\mathcal{E}_d$ is uncertain. Consider the machine example again where management wants to reduce the data storage for just one machine; however, as each machine results in standard deviations of differing sizes, it isn't obvious which machine would produce the most detrimental change to the uncertainty set. Thus, a transformation under structure 2 would indicate which machine, and therefore which data set, is the most sensitive to reductions in sample size and would pinpoint the worst machine to reduce the number the samples collected and would provide some measure by how much the sample size could be reduced before reaching robust infeasibility.

Definition 5.34. Let the RC distance to infeasibility under transformation structure 2 be:

$$\rho_2(d) = \inf_{\mathcal{E}_{\bar{d}} \cap \mathcal{I} \neq \emptyset} \{ \max_{j=1,\ldots,K} \alpha_j \mid \bar{d} = (P^0, \alpha_1 P^1, \ldots, \alpha_K P^K)\} - 1. \tag{5.35}$$

Let e be the vector of ones in $\mathbb{R}^K$ and $\alpha = (\alpha_1, \ldots, \alpha_K)^T$.

Since $\mathcal{E}_d \subset \mathcal{F}$, it is necessary to have at least one $\alpha_j \geq 1$ to obtain an infeasible robust problem instance, which guarantees $\rho_2(d) \geq 0$. Let $\alpha_{\max}$ be the component of α that attains the maximum in equation (5.35).

Rewriting the transformed RC having data $\bar{d} = (P^0, \alpha_1 P^1, \ldots, \alpha_K P^K)$ results in the equivalent SOCP with data

$$\bar{d} = \begin{bmatrix} \begin{bmatrix} P_1^0 \\ \operatorname{diag}(\alpha) R_1^T \end{bmatrix} \\ \vdots \\ \begin{bmatrix} P_m^0 \\ \operatorname{diag}(\alpha) R_m^T \end{bmatrix} \end{bmatrix}.$$

	Original	Perturbed
SOCP	$P_i^0 x - \Vert R_i^T x \Vert_2 \geq 0 \ \forall i$ $x \in C_X$	$P_i^0 x - \Vert \mathrm{diag}(\alpha) R_i^T x \Vert_2 \geq 0 \ \forall i$ $x \in C_X$
Data	$d = \begin{bmatrix} \tilde{R}_i^T \\ \vdots \\ \tilde{R}_i^T \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} P_1^0 \\ R_1^T \end{bmatrix} \\ \vdots \\ \begin{bmatrix} P_m^0 \\ R_m^T \end{bmatrix} \end{bmatrix}$	$\bar{d} = \begin{bmatrix} \begin{bmatrix} P_1^0 \\ \mathrm{diag}(\alpha) R_1^T \end{bmatrix} \\ \vdots \\ \begin{bmatrix} P_m^0 \\ \mathrm{diag}(\alpha) R_m^T \end{bmatrix} \end{bmatrix}$

Table 5.6: Changes to the SOCP data under structure II perturbation resulting from a structure 2 transformation on the RC data.

Notice that $\Delta_{II} d = \bar{d} - d$ corresponds to SOCP perturbation of structure II given in equation (5.21). Table 5.6 summarizes the changes to the SOCP data resulting from this perturbation.

Proposition 5.36. *Let the norm on the SOCP data $d \in \mathbb{R}^{m(K+1)\times n}$ be the matrix p-norm and the SOCP distance to infeasibility under perturbation structure II be defined as in equation* (5.23)*. Let the norm on the RC data be the matrix p-norm and the RC distance to infeasibility under transformation structure 2 be defined as in equation* (5.35)*. Then*

$$\rho_{II}(d) \leq \|S\|_p \rho_2(d).$$

Proof. If $\rho_2(d) = +\infty$, then the result is trivially true. Assume $\rho_2(d) < +\infty$. Let $\epsilon > 0$ and let $\alpha \in \mathbb{R}^K$ be such that $\mathcal{E}_{\bar{d}} \cap \mathcal{I} \neq \emptyset$, where $\bar{d} = (P^0, \alpha_1 P^1, \ldots, \alpha_K P^K)$ and $\alpha_{\max} - 1 \ \leq \epsilon + \rho_2(d)$.

Let $\Delta_{II} d$ be a perturbation of the SOCP data d having structure II obtained by replacing the appropriate blocks with blocks $\Delta R_i^T = (\mathrm{diag}(\alpha) - e) R_i^T, \ i = 1, \ldots, m$

in a $m(K+1)\times n$ matrix of zeros. Note that, since $\mathcal{E}_{\bar{d}} \cap \mathcal{I} \neq \emptyset$, $d+\Delta_{II}d \in \mathcal{I}_{\mathrm{SOCP}}$. Using the actual form of $\Delta_{II}d$ given in Table 5.6, we get the following bound:

$$\begin{aligned}
\|\Delta_{II}d\|_p &= \max_{\|x\|_p=1} \|\Delta_{II}dx\|_p \\
&= \max_{\|x\|_p=1} \|\mathrm{diag}(\mathcal{A}-E)R^T x\|_p \\
&= \|\mathrm{diag}(\mathcal{A}-E)R^T\|_p \\
&\leq \|\mathrm{diag}(\mathcal{A}-E)\|_p \|R^T\|_p \\
&= (\alpha_{\max}-1)\|R^T\|_p \\
&= (\alpha_{\max}-1)\|S\|_p.
\end{aligned}$$

where

$$\mathcal{A}-E = \begin{bmatrix} \alpha - e \\ \vdots \\ \alpha - e \end{bmatrix} \in \mathbb{R}^{Km}.$$

The second to last equality uses the result in equation (5.15). Hence,

$$\rho_{II}(d) \leq \|\Delta_{II}d\|_p,$$

which results in

$$\rho_{II}(d) \leq \|\Delta_{II}d\|_p \leq (\alpha_{\max}-1)\|S\|_p \leq \|S\|_p(\epsilon + \rho_2(d)).$$

Since the above is true for any $\epsilon > 0$, the desired result follows. $\square$

Notice that if we multiply each P^j by $\alpha_{\max}$, then we actually obtain a structure 1 transformation to the ellipsoid that corresponds to an infeasible instance of the RC (5.7); thus $\rho_1(d) \leq \rho_2(d)$. On the other hand, as a structure 2 transformation is more general than a structure 1 transformation, we have $\rho_2(d) \leq \rho_1(d)$. These two inequalities result in $\rho_1(d) = \rho_2(d)$.

5.6.3 Structured linear transformation of the ellipsoid: RC transformation structure 3

In this section we consider a more general transformation that changes the size and shape of the ellipsoid, but maintains the center P^0. In particular, we consider a transformation that results from pre-multiplying the matrices $P^1, \ldots, P^K$ by a matrix $Q \in \mathbb{R}^{m \times m}$, thus resulting in a linear transformation of the ellipsoid $\mathcal{E}_d$. We refer to this transformation as transformation of structure 3. Figure 5.5 illustrates an example of a structure 3 transformation, which shows that under a linear transformation, the axes of the ellipsoid are rotated and, in addition, their lengths change. Table 5.7 summarizes the changes to the original RC data and ellipsoid under transformation structure 3.

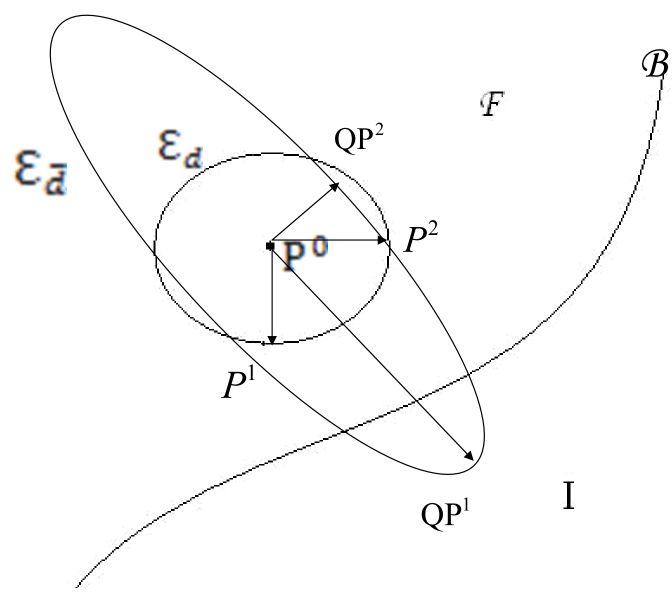


Figure 5.5: Example of RC transformation under structure 3 transformation; here $Q \in \mathbb{R}^{m \times m}$

To identify the data of the SOCP, equivalent to the transformed RC, first consider the following example where $m = 2$, $n = 3$, $K = 2$, and $Q \in \mathbb{R}^{2 \times 2}$.

	Original	Transformed
Data	$d = (P^0, P^1, \ldots, P^K)$	$\bar{d} = (P^0, QP^1, \ldots, QP^K)$
Ellipsoid	$\mathcal{E}_d = \{P^0 + \sum_{j=1}^{K} u_j P^j \mid \|u\|_2 \leq 1\}$	$\mathcal{E}_{\bar{d}} = \{P^0 + \sum_{j=1}^{K} u_j QP^j \mid \|u\|_2 \leq 1\}$

Table 5.7: Changes to the RC data under structure 3 transformation

$$
\begin{aligned}
\begin{bmatrix} QP^1 \\ QP^2 \end{bmatrix} &= \begin{bmatrix} \begin{bmatrix} q_{11} & q_{12} \\ q_{21} & q_{22} \end{bmatrix} \begin{bmatrix} P^1_{11} & P^1_{12} & P^1_{13} \\ P^1_{21} & P^1_{22} & P^1_{23} \end{bmatrix} \\ \begin{bmatrix} q_{11} & q_{12} \\ q_{21} & q_{22} \end{bmatrix} \begin{bmatrix} P^2_{11} & P^2_{12} & P^2_{13} \\ P^2_{21} & P^2_{22} & P^2_{23} \end{bmatrix} \end{bmatrix} \\
&= \begin{bmatrix} q_{11}P^1_{11} + q_{12}P^1_{21} & q_{11}P^1_{12} + q_{12}P^1_{22} & q_{11}P^1_{13} + q_{12}P^1_{23} \\ q_{21}P^1_{11} + q_{22}P^1_{21} & q_{21}P^1_{12} + q_{22}P^1_{22} & q_{21}P^1_{13} + q_{22}P^1_{23} \\ q_{11}P^2_{11} + q_{12}P^2_{21} & q_{11}P^2_{12} + q_{12}P^2_{22} & q_{11}P^2_{13} + q_{12}P^2_{23} \\ q_{21}P^2_{11} + q_{22}P^2_{21} & q_{21}P^2_{12} + q_{22}P^2_{22} & q_{21}P^2_{13} + q_{22}P^2_{23} \end{bmatrix} \\
&= \text{un-vec}\left[\begin{bmatrix} \begin{bmatrix} q_{11} & 0 & 0 & 0 & 0 & 0 \\ 0 & q_{11} & 0 & 0 & 0 & 0 \\ 0 & 0 & q_{11} & 0 & 0 & 0 \\ 0 & 0 & 0 & q_{11} & 0 & 0 \\ 0 & 0 & 0 & 0 & q_{11} & 0 \\ 0 & 0 & 0 & 0 & 0 & q_{11} \end{bmatrix} & \begin{bmatrix} q_{12} & 0 & 0 & 0 & 0 & 0 \\ 0 & q_{12} & 0 & 0 & 0 & 0 \\ 0 & 0 & q_{12} & 0 & 0 & 0 \\ 0 & 0 & 0 & q_{12} & 0 & 0 \\ 0 & 0 & 0 & 0 & q_{12} & 0 \\ 0 & 0 & 0 & 0 & 0 & q_{12} \end{bmatrix} \\ \begin{bmatrix} q_{21} & 0 & 0 & 0 & 0 & 0 \\ 0 & q_{21} & 0 & 0 & 0 & 0 \\ 0 & 0 & q_{21} & 0 & 0 & 0 \\ 0 & 0 & 0 & q_{21} & 0 & 0 \\ 0 & 0 & 0 & 0 & q_{21} & 0 \\ 0 & 0 & 0 & 0 & 0 & q_{21} \end{bmatrix} & \begin{bmatrix} q_{22} & 0 & 0 & 0 & 0 & 0 \\ 0 & q_{22} & 0 & 0 & 0 & 0 \\ 0 & 0 & q_{22} & 0 & 0 & 0 \\ 0 & 0 & 0 & q_{22} & 0 & 0 \\ 0 & 0 & 0 & 0 & q_{22} & 0 \\ 0 & 0 & 0 & 0 & 0 & q_{22} \end{bmatrix} \end{bmatrix} \begin{bmatrix} P^1_{11} \\ P^1_{12} \\ P^1_{13} \\ P^2_{11} \\ P^2_{12} \\ P^2_{13} \\ P^1_{21} \\ P^1_{22} \\ P^1_{23} \\ P^2_{21} \\ P^2_{22} \\ P^2_{23} \end{bmatrix} \right] \\
&= \text{un-vec}(\widehat{Q}\text{vec}(R^T)).
\end{aligned}
$$

In general, $\widehat{Q} \in \mathbb{R}^{Kmn \times Kmn}$ is the block-structured matrix consisting of elements of Q as defined below:

(5.37)

$$\widehat{Q} = \begin{bmatrix} \tilde{Q}_{11} & 0 & \dots & 0 & \tilde{Q}_{12} & 0 & \dots & 0 & \dots & \tilde{Q}_{1m} & 0 & \dots & 0 \\ 0 & \tilde{Q}_{11} & \dots & 0 & 0 & \tilde{Q}_{12} & \dots & 0 & \dots & 0 & \tilde{Q}_{1m} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \dots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \tilde{Q}_{11} & 0 & 0 & \dots & \tilde{Q}_{12} & \dots & 0 & 0 & \dots & \tilde{Q}_{1m} \\ \tilde{Q}_{21} & 0 & \dots & 0 & \tilde{Q}_{22} & 0 & \dots & 0 & \dots & \tilde{Q}_{2m} & 0 & \dots & 0 \\ 0 & \tilde{Q}_{21} & \dots & 0 & 0 & \tilde{Q}_{22} & \dots & 0 & \dots & 0 & \tilde{Q}_{2m} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \dots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \tilde{Q}_{21} & 0 & 0 & \dots & \tilde{Q}_{22} & \dots & 0 & 0 & \dots & \tilde{Q}_{2m} \\ & & \vdots & & & & \vdots & & & & \vdots & & \\ \tilde{Q}_{m1} & 0 & \dots & 0 & \tilde{Q}_{m2} & 0 & \dots & 0 & \dots & \tilde{Q}_{mm} & 0 & \dots & 0 \\ 0 & \tilde{Q}_{m1} & \dots & 0 & 0 & \tilde{Q}_{m2} & \dots & 0 & \dots & 0 & \tilde{Q}_{mm} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \dots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \tilde{Q}_{m1} & 0 & 0 & \dots & \tilde{Q}_{m2} & \dots & 0 & 0 & \dots & \tilde{Q}_{mm} \end{bmatrix},$$

and

$$\tilde{Q}_{ab} = \begin{bmatrix} q_{ab} & 0 & \dots & 0 \\ 0 & q_{ab} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & q_{ab} \end{bmatrix} \in \mathbb{R}^{n \times n}$$

where q_{ab} is the $(a, b)^{th}$ element of Q.

Let $\tilde{Q}_i \in \mathbb{R}^{Kn \times Kmn}$ denote the i^{th}-row block of $\widehat{Q}$:

$$\tilde{Q}_i = \begin{bmatrix} \tilde{Q}_{i1} & 0 & \dots & 0 & \tilde{Q}_{i2} & 0 & \dots & 0 & \dots & \tilde{Q}_{im} & 0 & \dots & 0 \\ 0 & \tilde{Q}_{i1} & \dots & 0 & 0 & \tilde{Q}_{i2} & \dots & 0 & \dots & 0 & \tilde{Q}_{im} & \dots & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots & \dots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \tilde{Q}_{i1} & 0 & 0 & \dots & \tilde{Q}_{i2} & \dots & 0 & 0 & \dots & \tilde{Q}_{im} \end{bmatrix},$$

then rewriting the transformed RC having data $\bar{d} = (P^0, QP^1, \dots, QP^K)$ results in the equivalent SOCP with data

$$\bar{d} = \begin{bmatrix} \begin{bmatrix} P_1^0 \\ \text{un-vec}(\tilde{Q}_1 \text{vec}(R^T)) \end{bmatrix} \\ \vdots \\ \begin{bmatrix} P_m^0 \\ \text{un-vec}(\tilde{Q}_m \text{vec}(R^T)) \end{bmatrix} \end{bmatrix}.$$

Notice, $\Delta_{II} d = \bar{d} - d$ corresponds to SOCP perturbation of structure II given in equation (5.21). Table 5.8 summarizes the changes to the SOCP data resulting from this perturbation.

	Original	Perturbed
SOCP	$P_i^0 x - \lVert R_i^T x \rVert_2 \geq 0 \ \forall i$ $x \in C_X$	$P_i^0 x - \lVert \text{un-vec}(\tilde{Q}_i \text{vec}(R^T))x \rVert_2 \geq 0 \ \forall i$ $x \in C_X$
Data	$d = \begin{bmatrix} \tilde{R}_i^T \\ \vdots \\ \tilde{R}_i^T \end{bmatrix} = \begin{bmatrix} \begin{bmatrix} P_1^0 \\ R_1^T \end{bmatrix} \\ \vdots \\ \begin{bmatrix} P_m^0 \\ R_m^T \end{bmatrix} \end{bmatrix}$	$\bar{d} = \begin{bmatrix} \begin{bmatrix} P_1^0 \\ \text{un-vec}(\tilde{Q}_1 \text{vec}(R^T)) \end{bmatrix} \\ \vdots \\ \begin{bmatrix} P_m^0 \\ \text{un-vec}(\tilde{Q}_m \text{vec}(R^T)) \end{bmatrix} \end{bmatrix}$

Table 5.8: Changes to the SOCP data under structure II perturbation resulting from a structure 3 transformation on the RC data.

Definition 5.38. Let the RC distance to infeasibility under transformation structure 3 be:

$$\rho_3(d) = \inf_{\mathcal{E}_{\bar{d}} \cap \mathcal{I} \neq \emptyset} \{ \lVert \widehat{Q} - I_{Kmn} \rVert \mid \bar{d} = (P^0, QP^1, \ldots, QP^K) \} \tag{5.39}$$

where $Q \in \mathbb{R}^{m \times m}$, I_{Kmn} is the $Kmn \times Kmn$ identity matrix and $\widehat{Q}$ is defined in (5.37).

Proposition 5.40. *Let the norm on the SOCP data $d \in \mathbb{R}^{m(K+1) \times n}$ be the matrix p-norm and the SOCP distance to infeasibility under perturbation structure II be defined as in equation* (5.23)*. Let the norm on the RC data be the matrix p-norm and the RC distance to infeasibility under transformation structure 3 be defined as in equation* (5.39)*. Then*

$$\rho_{II}(d) \leq \lVert S \rVert_p \rho_3(d). \tag{5.41}$$

Proof. If $\rho_3(d) = +\infty$, then the result is trivially true. Assume $\rho_3(d) < +\infty$. Let $\epsilon > 0$ and let Q be such that $\mathcal{E}_{\bar{d}} \cap \mathcal{I} \neq \emptyset$, where $\bar{d} = (P^0, QP^1, \ldots, QP^K)$ and $\lVert \widehat{Q} - I_{Kmn} \rVert_p \leq \epsilon + \rho_3(d)$.

Let $\Delta_{II}d$ be a perturbation of the SOCP data d having structure II obtained by replacing the appropriate blocks with blocks $\Delta R_i^T = \text{un-vec}(\tilde{Q}_i \text{vec}(R^T)) - R_i^T,\ i = 1, \ldots, m$ in a $m(K+1) \times n$ matrix of zeros. Note that, since $\mathcal{E}_{\bar{d}} \cap \mathcal{I} \neq \emptyset$, $d + \Delta_{II}d \in \mathcal{I}_{\text{SOCP}}$. Using the actual form of $\Delta_{II}d$ given in Table 5.8, we get the following bound:

$$
\begin{aligned}
\|\Delta_{II}d\|_p &= \max_{\|x\|_p=1} \|\Delta_{II}dx\|_p \\
&= \max_{\|x\|_p=1} \|(\text{un-vec}(\widehat{Q}\text{vec}(R^T)) - R^T)x\|_p \\
&= \max_{\|x\|_p=1} \|(\text{un-vec}((\widehat{Q} - I_{Kmn})\text{vec}(R^T)))x\|_p \\
&= \|\text{un-vec}((\widehat{Q} - I_{Kmn})\text{vec}(R^T))\|_p \\
&\leq \|(\widehat{Q} - I_{Km})\|_p \|R^T\|_p \\
&= \|(\tilde{Q} - I_{Km})\|_p \|S\|_p.
\end{aligned}
$$

Hence,

$$\rho_{II}(d) \leq \|\Delta_{II}d\|_p,$$

which results in

$$\rho_{II}(d) \leq \|\Delta_{II}d\|_p \leq \|\tilde{Q} - I_{Km}\|_p \|S\|_p \leq \|S\|_p(\epsilon + \rho_3(d)).$$

Since the above is true for any $\epsilon > 0$, the desired result follows. $\square$

In conclusion, what we have been able to show is that if the RC is not well-posed ($\mathcal{E}_d$ is nearly infeasible), then neither is the SOCP, given that the distance to infeasibility of the SOCP is bounded above by an infeasible perturbation to the RC. One interesting research hypothesis that remains to be proved (or disproved) is the following: if the SOCP is not well-posed, then neither is the RC. Proving this statement would result in an if and only if condition concerning the well-posedness of the RC and SOCP.

CHAPTER VI

Conclusions

In Chapters II, III, and IV we have proposed an approach to linear optimization with recourse that is robust with respect to the underlying probabilities. Specifically, instead of relying on the actual distribution, which would be difficult to estimate accurately, or a family of distributions, which would significantly increase the complexity of the problem at hand, we have modeled random variables as uncertain parameters in a polyhedral uncertainty set and analyzed the problem for the worst-case instance within that set. We have shown that this robust formulation can be solved using a cutting-plane algorithm and standard linear optimization software. We tested our approach on a multi-item newsvendor problem and a production planning problem with demand uncertainties, with encouraging computational results. Analysis of obtained solutions provides insight into appropriate levels of conservatism in planning (as captured by the budget of uncertainty) to obtain lower average costs.

Recently cutting-plane algorithms have been used to solve adjustable robust optimization problems, but the work focused on solving linear adjustable robust problems. An obvious next step is to use cutting-plane algorithms to solve nonlinear adjustable robust optimization problems and see if Kelley's method is still superior, in computational performance, to other cutting-plane algorithms such as analytic center or the subgradient algorithm. As the computational tractability of the cutting-plane algorithm approach is dependent upon the uncertainty set, the impact of various uncertainty sets on the computational tractability should be investigated.

In Chapter V we consider a robust counterpart of a linear feasibility problem subjected to data perturbations characterized by an ellipsoidal uncertainty set, and propose several possible measures of distance to ill-posedness for such a problem. We connect these measures to the size and shape of the uncertainty set used to define the robust problem, and provide bounds on these measures in terms of the traditional structured distance to ill-posedness of an equivalent conic feasibility problem, thus grounding our ideas in established concepts. One interesting research hypothesis remains unanswered concerning the connection between the well-posedness of the RC and the SOCP: given that we have shown if the RC is not well-posed ($\mathcal{E}_d$ is nearly infeasible), then neither is the SOCP, does the converse hold, i.e., if the SOCP is not well-posed, then neither is the RC? Addressing this question would give us insight into the connection between the conditioning of the two problems and seems the first step in the continuation of this research.

The eventual goal of this line of research though is to develop measures of conditioning for robust optimization problems that would be helpful in understanding the interplay between the choice of the uncertainty set and the impact of that choice on the structure and objective function value of the resulting robust problem, and provide modeling guidance in selecting uncertainty sets that are appropriate for the problem at hand. Chapter V only presents measures of distance to ill-posedness for a robust problem under ellipsoidal uncertainty; however, there are equivalent conic formulations for robust counterparts characterized by other types of uncertainty sets and the impact of alternative uncertainty sets should be investigated. Moreover, the results presented concerned a robust linear feasibility problem; other types of programs (feasibility and optimization problems) should be considered.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Alper Atamtürk and Muhong Zhang. Two-stage robust network flow and design under demand uncertainty. *Oper. Res.*, 55(4):662–673, 2007.

[2] D. Atkinson and P. Vaidya. A cutting plane algorithm for convex programming that uses analytic centers. *Math. Programming*, 69(1, Ser. B):1–43, 1995. Nondifferentiable and large-scale optimization (Geneva, 1992).

[3] Igor Averbakh and Yun-Bin Zhao. Explicit reformulations for robust optimization problems with general uncertainty sets. *SIAM J. Optim.*, 18(4):1436–1466, 2007.

[4] O. Bahn, O. du Merle, J.-L. Goffin, and J.-P. Vial. A cutting plane method from analytic centers for stochastic programming. *Math. Programming*, 69(1, Ser. B):45–73, 1995. Nondifferentiable and large-scale optimization (Geneva, 1992).

[5] Mokhtar S. Bazaraa, Hanif D. Sherali, and C. M. Shetty. *Nonlinear programming*. Wiley-Interscience [John Wiley & Sons], Hoboken, NJ, third edition, 2006. Theory and algorithms.

[6] A. Ben-Tal, A. Goryashko, E. Guslitzer, and A. Nemirovski. Adjustable robust solutions of uncertain linear programs. *Math. Program.*, 99(2, Ser. A):351–376, 2004.

[7] A. Ben-Tal and A. Nemirovski. Robust truss topology design via semidefinite programming. *SIAM J. Optim.*, 7(4):991–1016, 1997.

[8] A. Ben-Tal and A. Nemirovski. Robust convex optimization. *Math. Oper. Res.*, 23(4):769–805, 1998.

[9] A. Ben-Tal and A. Nemirovski. Robust solutions of uncertain linear programs. *Oper. Res. Lett.*, 25(1):1–13, 1999.

[10] A. Ben-Tal and A. Nemirovski. Robust solutions of linear programming problems contaminated with uncertain data. *Math. Program.*, 88(3, Ser. A):411–424, 2000.

[11] A. Ben-Tal and A. Nemirovski. On approximate robust counterparts of uncertain semidefinite and conic quadratic programs. In *System modeling and optimization, XX (Trier, 2001)*, volume 130 of *IFIP Int. Fed. Inf. Process.*, pages 1–22. Kluwer Acad. Publ., Boston, MA, 2003.

[12] A. Ben-Tal, A. Nemirovski, and C. Roos. Robust solutions of uncertain quadratic and conic quadratic problems. *SIAM J. Optim.*, 13(2):535–560 (electronic), 2002.

[13] J. F. Benders. Partitioning procedures for solving mixed-variables programming problems. *Comput. Manag. Sci.*, 2(1):3–19, 2005. Reprinted from Numer. Math. **4** (1962), 238–252 [MR0147303].

[14] Dimitris Bertsimas and Melvyn Sim. Robust discrete optimization and network flows. *Math. Program.*, 98(1-3, Ser. B):49–71, 2003. Integer programming (Pittsburgh, PA, 2002).

[15] Dimitris Bertsimas and Melvyn Sim. The price of robustness. *Oper. Res.*, 52(1):35–53, 2004.

[16] Dimitris Bertsimas and Melvyn Sim. Tractable approximations to robust conic quadratic optimization problems. *Math. Program.*, 107(1-2):5–36, 2006.

[17] Dimitris Bertsimas and Aurélie Thiele. A robust optimization approach to supply chain management. In *Integer programming and combinatorial optimization*, volume 3064 of *Lecture Notes in Comput. Sci.*, pages 86–100. Springer, Berlin, 2004.

[18] Dimitris Bertsimas and Aurélie Thiele. A robust optimization approach to inventory theory. *Oper. Res.*, 54(1):150–168, 2006.

[19] Dimitris Bertsimas and John N. Tsitsiklis. *Introduction to linear optimization.* Athena Scientific and Dynamic Ideas, Belmont, Massachusetts, 1997.

[20] Daniel Bienstock. Experiments in robust portfolio optimization. Technical report, Columbia University, January 2007. Columbia Center for Financial Engineering Report 2007-01.

[21] Daniel Bienstock and Nuri Özbay. Computing robust basestock levels. *Discrete Optim.*, 5(2):389–414, 2008.

[22] John R. Birge and François Louveaux. *Introduction to stochastic programming.* Springer Series in Operations Research. Springer-Verlag, New York, 1997.

[23] Dennis Cheung and Felipe Cucker. A new condition number for linear programming. *Math. Program.*, 91(1, Ser. A):163–174, 2001.

[24] Dennis Cheung and Felipe Cucker. Solving linear programs with finite precision. I. Condition numbers and random programs. *Math. Program.*, 99(1, Ser. A):175–196, 2004.

[25] Dennis Cheung and Felipe Cucker. Solving linear programs with finite precision. II. Algorithms. *J. Complexity*, 22(3):305–335, 2006.

[26] M. Chu, Y. Zinchenko, S. Henderson, and M. Sharpe. Robust optimization for intensity modulated radiation therapy treatment planning under uncertainty. *Physics in Medicine and Biology*, 50(23):5463–5477, 2005.

[27] George B. Dantzig. Linear programming under uncertainty. *Management Sci.*, 1:197–206, 1955.

[28] Laurent El Ghaoui and Hervé Lebret. Robust solutions to least-squares problems with uncertain data. *SIAM J. Matrix Anal. Appl.*, 18(4):1035–1064, 1997.

[29] Laurent El Ghaoui, Francois Oustry, and Hervé Lebret. Robust solutions to uncertain semidefinite programs. *SIAM J. Optim.*, 9(1):33–52 (electronic), 1999.

[30] Marina Epelman and Robert M. Freund. Condition number complexity of an elementary algorithm for computing a reliable solution of a conic linear system. *Math. Program.*, 88(3, Ser. A):451–485, 2000.

[31] Marina Epelman and Robert M. Freund. A new condition measure, preconditioners, and relations between different measures of conditioning for conic linear systems. *SIAM J. Optim.*, 12(3):627–655 (electronic), 2002.

[32] Sharon Filipowski. On the complexity of solving feasible systems of linear inequalities specified with approximate data. *Math. Programming*, 71(3, Ser. A):259–288, 1995.

[33] Sharon Filipowski. On the complexity of solving sparse symmetric linear programs specified with approximate data. *Math. Oper. Res.*, 22(4):769–792, 1997.

[34] Robert M. Freund and Fernando Ordóñez. On an extension of condition number theory to nonconic convex optimization. *Math. Oper. Res.*, 30(1):173–194, 2005.

[35] Robert M. Freund, Fernando Ordóñez, and Kim-Chuan Toh. Behavioral measures and their correlation with IPM iteration counts on semi-definite programming problems. *Math. Program.*, 109(2-3, Ser. B):445–475, 2007.

[36] Robert M. Freund and Jorge R. Vera. Condition-based complexity of convex optimization in conic linear form via the ellipsoid algorithm. *SIAM J. Optim.*, 10(1):155–176 (electronic), 1999.

[37] Robert M. Freund and Jorge R. Vera. Some characterizations and properties of the "distance to ill-posedness" and the condition measure of a conic linear system. *Math. Program.*, 86(2, Ser. A):225–260, 1999.

[38] Robert M. Freund and Jorge R. Vera. On the complexity of computing estimates of condition measures of a conic linear system. *Math. Oper. Res.*, 28(4):625–648, 2003.

[39] J. Goffin, Z. Luo, and Y. Ye. Complexity analysis of an interior cutting plane method for convex feasibility problems. *SIAM J. Optim.*, 6(3):638–652, 1996.

[40] D. Goldfarb and G. Iyengar. Robust convex quadratically constrained programs. *Math. Program.*, 97(3, Ser. B):495–515, 2003. New trends in optimization and computational algorithms (NTOC 2001) (Kyoto).

[41] D. Goldfarb and G. Iyengar. Robust portfolio selection problems. *Math. Oper. Res.*, 28(1):1–38, 2003.

[42] Peter Kall and Stein W. Wallace. *Stochastic programming*. Wiley-Interscience Series in Systems and Optimization. John Wiley & Sons Ltd., Chichester, 1994.

[43] J. E. Kelley, Jr. The cutting-plane method for solving convex programs. *J. Soc. Indust. Appl. Math.*, 8:703–712, 1960.

[44] Carl Meyer. *Matrix Analysis and Applied Linear Algebra*. Society for Industrial and Applied Mathematics, 2000.

[45] J. Mitchell. Computational experience with an interior point cutting plane algorithm. *SIAM J. Optim.*, 10(4):1212–1227 (electronic), 2000.

[46] J. Mitchell. Polynomial interior point cutting plane methods. *Optim. Methods Softw.*, 18(5):507–534, 2003.

[47] Almir Mutapcic and Stephen Boyd. Cutting-set methods for robust convex optimization with pessimizing oracles, 2008.

[48] Y. Nesterov. Complexity estimates of some cutting plane methods based on the analytic barrier. *Math. Programming*, 69(1, Ser. B):149–176, 1995. Nondifferentiable and large-scale optimization (Geneva, 1992).

[49] Manuel A. Nunez and Robert M. Freund. Condition measures and properties of the central trajectory of a linear program. *Math. Programming*, 83(1, Ser. A):1–28, 1998.

[50] Fernando Ordóñez and Robert M. Freund. Computational experience and the explanatory value of condition measures for linear optimization. *SIAM J. Optim.*, 14(2):307–333 (electronic), 2003.

[51] Fernando Ordóñez and Jiamin Zhao. Robust capacity expansion of network flows. *Networks*, 50(2):136–145, 2007.

[52] J. Peña and J. Renegar. Computing approximate solutions for convex conic systems of constraints. *Math. Program.*, 87(3, Ser. A):351–383, 2000.

[53] Javier Peña. Understanding the geometry of infeasible perturbations of a conic linear system. *SIAM J. Optim.*, 10(2):534–550 (electronic), 2000.

[54] Javier Peña. Conditioning of convex programs from a primal-dual perspective. *Math. Oper. Res.*, 26(2):206–220, 2001.

[55] Javier Peña. A characterization of the distance to infeasibility under block-structured perturbations. *Linear Algebra Appl.*, 370:193–216, 2003.

[56] Javier Peña. On the block-structured distance to non-surjectivity of sublinear mappings. *Math. Program.*, 103(3, Ser. A):561–573, 2005.

[57] James Renegar. Some perturbation theory for linear programming. *Math. Programming*, 65(1, Ser. A):73–91, 1994.

[58] James Renegar. Incorporating condition measures into the complexity theory of linear programming. *SIAM J. Optim.*, 5(3):506–524, 1995.

[59] James Renegar. Linear programming, complexity theory and elementary functional analysis. *Math. Programming*, 70(3, Ser. A):279–351, 1995.

[60] James Renegar. Condition numbers, the barrier method, and the conjugate-gradient method. *SIAM J. Optim.*, 6(4):879–912, 1996.

[61] A. L. Soyster. Convex programming with set-inclusive constraints and applications to inexact linear programming. *Oper. Res.*, 21(5):1154–1157, 1973.

[62] A. Takeda, S. Taguchi, and R. H. Tütüncü. Adjustable robust optimization models for a nonlinear two-period system. *J. Optim. Theory Appl.*, 136(2):275–295, 2008.

[63] Aurélie Thiele, Tara Terry, and Marina Epelman. Robust linear optimization with recourse. Technical report, University of Michigan, Industrial and Operations Engineering, March 2009.

[64] Jorge R. Vera. Ill-posedness and the complexity of deciding existence of solutions to linear programs. *SIAM J. Optim.*, 6(3):549–569, 1996.

[65] Jorge R. Vera. On the complexity of linear programming under finite precision arithmetic. *Math. Program.*, 80(1):91–123, 1998.

[66] Laurence A. Wolsey. *Integer programming*. Wiley-Interscience Series in Discrete Mathematics and Optimization. John Wiley & Sons Inc., New York, 1998. A Wiley-Interscience Publication.