Multi-Period Pricing for Perishable Products: Uncertainty and Competition

by

Lei Zhang

B.S. Computer Science, National University of Singapore (2003)

Submitted to the School of Engineering
in partial fulfillment of the requirements for the degree of

Master of Science in Computation for Design and Optimization

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2006

Author .....
School of Engineering
August 11, 2006

Certified by .....
Georgia Perakis
J. Spencer Standish Associate Professor of Operations Research
Thesis Supervisor

Accepted by .....
Jaime Peraire
Professor of Aeronautics and Astronautics
Codirector, Computation for Design and Optimization Program

Multi-Period Pricing for Perishable Products: Uncertainty and Competition

by

Lei Zhang

Submitted to the School of Engineering
on August 11, 2006, in partial fulfillment of the
requirements for the degree of
Master of Science in Computation for Design and Optimization

Abstract

The pricing problem in a multi-period setting is a challenging problem and has attracted much attention in recent years. In this thesis, we consider a monopoly and an oligopoly pricing problem. In the latter, several sellers simultaneously seek an optimal pricing policy for their products. The products are assumed to be differentiated and substitutable. Each seller has the option to set prices for her products at each time period, and her goal is to find a pricing policy that will yield the maximum overall profit. Each seller has a fixed initial inventory of each product to be allocated over the entire time horizon and does not have the option to produce additional inventory between periods. There are no holding costs or back-order costs. In addition, the products are perishable and have no salvage costs. This means that at the end of the entire time horizon, any remaining products will be worthless. The demand function each seller faces for each product is uncertain and is affected by both the prices at the current period and past pricing history for her and her competitors.

In this thesis, we address both the uncertain and the competitive aspect of the problem. First, we study the uncertain aspect of the problem in a simplified setting, where there is only one seller and two periods in the model. We use ideas of robust optimization, adjustable robust optimization, dynamic programming and stochastic optimization to find adaptable closed loop pricing policies. Theoretical and numerical results show how the budget of uncertainty, the presence of a reference price, delayed resource allocation, and feedback control affect the quality of the pricing policies. Second, we extend the model to a multi-period setting, where the computation becomes a major issue. We use a delayed constraint generation method to significantly increase the size of the problem that our models can handle. Finally, we consider the pricing problem in an oligopoly setting. We show the existence of solution for both the best response subproblem and the market equilibrium problem for all of the models we discuss in the thesis. We also consider an iterative learning algorithm and illustrate through simulations that an equilibrium pricing policy can be computed for all of our models.

Thesis Supervisor: Georgia Perakis
Title: J. Spencer Standish Associate Professor of Operations Research

Acknowledgments

First, I would like to express my deepest gratitude to my research advisor, Prof. Georgia Perakis. She has shown incredible support and patience for my research. Moreover, she is exceptionally responsible to her students. She always gave me prompt replies to whatever questions I have asked. Even when she was on her sabbatical leave or attending a conference, she still wanted to talk to me over the phone at least weekly to know my progress and clear my doubts. I am so glad to have had her as my research advisor. Without her, I would not have been able to complete my research within such a short time. More importantly, she has given me a wide exposure to various aspects of the pricing problem, and I have learned a lot under her guidance.

Secondly, I would like to thank my fellow students in the CDO program. We have shared all the joys and pains in the past year. We spent many hours working together in the lab on the project and research. They have made the past year much more meaningful to me.

Thirdly, I would like to thank the staff at the Operations Research Center and the SMA office. They have made my stay at MIT a wonderful experience!

Finally, I would like to thank my family for their endless love and support in my life. They have made me strong and everything I am today!

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivations

In recent years, there has been a rising interest in revenue management in many industries such as the airline, the supply chain, and the retail industry. Many firms have realized that the profitability of a firm is critically affected by decisions such as pricing and inventory control. They invest more resources or even hire specialized consultants to help them to optimize their revenue management. Pricing is a critical factor in revenue management. A study by McKinsey and Company on the cost structure of Fortune 1000 companies in 2001 ([2]) shows that pricing is a more powerful lever than variable cost, fixed cost or sales improvement. This reveals that an improvement of 1% in pricing will yield an average of 8.6% in operating margin improvement. Therefore, a firm's ability to design a good pricing model will determine its performance in today's competitive market.

1.1.1 Demand uncertainty

The traditional approach to the dynamic pricing problem assumes that the input data is known exactly (that is, it always takes a nominal value). However, uncertainty is inherent in nature and any forecast into the future will involve a certain level of randomness. In addition, the demand model may not represent the relationship between

demand and price accurately. In fact, any demand model is just an approximation of the true relationship between demand and price, as there are many more factors that exist in reality and are not considered in this relationship in order to simplify the modeling. Furthermore, even if the demand model is exact, it is still difficult to determine the exact values of its parameters. One way to determine these parameters is to learn them from the historical data. Still, it just gives an estimation of the parameters, as the future cannot be inferred from the past in general.

One possible way of dealing with uncertain parameters is to assume that they follow a certain probability distribution. With this assumption, dynamic programming (DP) and stochastic optimization can be used to maximize the expected overall profit effectively. DP is an attractive and powerful technique to address problems with uncertainty. It models the overall dynamic decision process as a sequence of simpler optimization problems. This feature makes it possible to reveal the theoretical structure of the optimal policy for simple systems. However, DP also has its limitations. The main drawback is that the complexity of the underlying recursive optimization problem can explode with the number of state variables. This phenomenon is commonly known as the "curse of dimensionality."

Stochastic optimization is another leading approach to problems with uncertainty. Stochastic optimization takes advantage of the known probability distribution and finds a pricing policy that maximizes the expected overall profit. The strength of stochastic optimization is that recourse decisions can be made in the later stages in order to compensate for any bad effects that might have been experienced as a result of the decisions made in the earlier stages. Furthermore, stochastic optimization does not suffer from the curse of dimensionality in the state variables. However, it is limited to solving problems with few periods, as the number of possible scenarios that it considers grows exponentially with the number of periods.

The assumption made by dynamic programming and stochastic optimization on the probability distribution of the uncertain parameters being known in advance is controversial. Similar to the problem in determining the exact value for the parameters in the demand model, many researchers also question the validity of determining

the probability distribution from historical data. Thus, the need arises for a new optimization methodology that can address uncertainty without making specific assumptions on a probability distribution and that is computationally tractable.

Robust optimization has emerged as another popular approach to handling problems with uncertainty. Contrary to DP and stochastic optimization, it does not need to assume any probability distribution on the uncertain parameters. The only assumption it requires is that the uncertain parameters reside within a deterministic uncertainty set with some known nominal values. Robust optimization adopts a min-max approach to maximize the objective value in the worst-case scenario. It addresses data uncertainty by guaranteeing the feasibility and optimality of the solution against all possible instances of the parameters within the uncertainty set. Robust optimization is also computationally attractive, and it has been successfully applied to some large-scale and highly complex engineering problems. However, as it is maximizing the objective value in the worst-case scenario, it has also been criticized as being too conservative. To address this problem several robust optimization approaches have been proposed recently. These include adjusting the level of conservativeness through the notion of budget of uncertainty ([7]) or incorporating the idea of feedback control into the robust model ([4]).

1.1.2 Oligopoly and competition

In a monopoly setting, each seller's objective is to find an optimal pricing policy subject to her resource constraints. Her decision is not affected by other sellers' strategies in the market. The pricing problem is essentially a constrained optimization problem. However, in an oligopoly setting, competition among sellers arises, and one seller's pricing policy is influenced by the pricing policies of other sellers in the market. The fact that the pricing problem involves considerations about the pricing policies of competitors causes these models to take the form of a game. A game-theoretic framework may be required to study the pricing problem under these circumstances.

There are many ways of modeling competition in an oligopoly setting. For example, sellers in the market can be cooperative and work together to achieve a global

optimal solution of the whole system. In this case, a seller may sacrifice her own profit in order to work towards a global optimal solution. The strength of this model is that it is able to achieve a better global solution. However, this is usually at the cost of some sellers' own benefits. How to ensure that policies are fair to every seller so that they are willing to stay cooperative in the game is an issue that needs to be addressed in such a setting. In contrast, sellers can also be selfish by maximizing their own profits without worrying about the global performance of the system. In this model, a global optimal solution is unlikely to be obtained. However, fairness can be easily achieved, as each seller is free to make her own decision and not forced to sacrifice her benefit for others. This model is perhaps more realistic compared to the previous model.

In this thesis, we consider an oligopoly setting with non-cooperative sellers, as we believe it is a better model of reality. Each seller competes with other sellers in the market and seeks to optimize her own profits.

1.1.3 Closed-loop pricing policy

In an open-loop framework, the value of data over the entire time horizon is known in advance. Under such a setting, each seller is able to find an optimal pricing policy at time zero and commit to her decisions over the time horizon. However, when the value of data is unknown or allowed to vary over time, an open-loop pricing policy may perform very poorly, as it cannot adjust its decisions with respect to the changes in the system. In contrast, in a closed-loop pricing policy, the decisions for prices can be postponed until the last possible moment. It takes advantage of the fact that not all decisions have to be made at time zero. For example, the price for the last time period can be made at any time before the start of the last period. By postponing the decisions for prices until the last possible moment, sellers are able to collect more information about data and therefore, adjust their decisions to yield a higher overall profit.

A closed-loop pricing policy is not interested in finding the optimal price for each period but rather an optimal rule for setting a price for each period based on the

information that is available at that moment. For example, DP generates a look-up table of optimal prices for each value of the state variable. Stochastic optimization finds the optimal prices for every scenario that may occur. The robust optimization paradigm has been recently extended to incorporate this aspect. This is termed as affinely adjustable robust counterpart (AARC). The main feature in AARC is that some of the decisions can be made after a portion of the uncertain data is realized.

1.2 Literature review

There is a huge literature on revenue management and pricing. In [23], Talluri and van Ryzin provide an overview of the revenue management and pricing literature. In this section, we focus on literature on three aspects of the pricing problem. First, we review research that handles the uncertainty aspect in the pricing model. Then we discuss papers related to oligopolistic competition. Finally, we discuss literature relevant to the closed-loop policy and the notion of a reference price.

1.2.1 Demand uncertainty

The problem of demand uncertainty has motivated a significant amount of literature in the field of revenue management. Various models have been introduced to model demand uncertainty. In [26], Zabel introduces two models of demand uncertainty: a multiplicative and an additive demand model. In the multiplicative model, the demand at time t, denoted by d_t, is defined as $d_t = \eta_t u(p_t)$, where p_t is the price at period t, η_t is the uncertain factor that is assumed to follow either an exponential or a uniform distribution with $E[\eta_t] \geq 0$, and $u(p_t)$ is a decreasing function of p_t. In contrast, in the additive model, d_t is defined as $d_t = u(p_t) + \eta_t$.

In the Operations Research literature, there are several different ways of treating demand uncertainty. For example, the uncertain parameters are sometimes assumed to be deterministic first, and subsequently a sensitive analysis is performed to study the stability of the solution with respect to the small perturbations of the parameters. When the probability distribution of the underlying uncertain parameters is known,

stochastic optimization can be used to find a solution that either has a high expected objective value or a low constraint violation. DP is also commonly used to handle demand uncertainty, and it seeks to find an optimal solution for every value of the state variable.

Unlike stochastic optimization or DP, robust optimization does not assume any probability distribution of the uncertain parameters. Instead, it only requires the parameters to reside within an uncertainty set with some known nominal values, and uses a min-max approach to find an optimal solution in the worst-case scenario. Robust optimization was first considered by Soyster ([22]) in a linear optimization problem, where the data is uncertain within a convex set. This work adopts a worst-case approach, which, as a result, significantly decreased the performance of the solution. To address the problem of over-conservativeness, Ben-Tal and Nemirovski ([5],[6]) consider an ellipsoidal uncertainty set. This approach applies to linear programming and general convex programming. They show that the robust counterpart of many convex optimization problems with data within an ellipsoidal uncertainty set can be solved exactly or approximatively by polynomial-time algorithms. However, the transformation required for this type of uncertainty set is complex. The robust counterpart of a linear programming problem is reformulated as a second-order cone programming (SOCP), and a SOCP is then reformulated as a semidefinite programming (SDP), and the robust counterpart of SDP is NP-hard to solve. Bertsimas and Sim ([7]) introduce a robust optimization model in which the robust counterpart of linear optimization remains a linear optimization problem. They also introduce an attractive way of adjusting the level of conservativeness through the notation of a budget of uncertainty.

The robust optimization paradigm has been applied to a number of areas such as the pricing problem. Perakis and Sood ([17]) and Perakis and Nguyen ([18]) study a multi-period oligopolistic market for a perishable product setting with demand uncertainty. They address competition and demand uncertainty using ideas from robust optimization and quasi-variational inequalities. Nevertheless, the lack of feedback control in this model can make the robust approach overly conservative. In [15],

Goldfarb and Iyengar apply robust optimization to portfolio selection problems. They introduce an uncertainty set that allows them to reformulate the robust counterpart as SOCP.

1.2.2 Competition

Oligopolistic competition in the field of pricing has become a popular topic to research in recent years. In [25], Vives presents several pricing models in an oligopoly market. Fudenberg and Tirole ([12]) review a number of game theoretic models for pricing and capacity decisions. Gaimon ([14]) studies both the open- and closed-loop Nash equilibria for two firms and a single product setting, where the price and capacity are determined to maximize the net profits. Kirman and Sobel ([16]) develop a multi-period model of oligopoly with random demand. They show the existence of equilibrium pricing strategies for the firm. In [20], Rosen shows existence of the equilibrium solutions under concavity of the payoff to a seller with respect to its own strategy space and convexity of the joint strategy space. Uniqueness is shown under strict diagonal dominance of the Hessian matrix of the payoff function. Perakis and Sood ([17]) and Sood ([21]) study the competitive multi-period pricing problem for a single perishable product, while Perakis and Nguyen ([18]) study the same problem for many perishable products sharing capacity. They use results from variational inequality theory and robust optimization to establish the existence of the pricing equilibrium policy and comment on the uniqueness of the pricing equilibrium policy. Perakis and Adida ([1],[17]) present a continuous time optimal control model for studying a dynamic pricing and inventory control problem with no back-orders. In many cases, the equilibrium point is obtained by solving the differential form of the Kuhn-Tucker optimality conditions in these applications.

Quasi-variational inequalities in finite dimension have been widely studied to model dynamic systems of conflict and cooperation, where the decisions are made over a time horizon. Cavazzuti and others ([8]) introduce some relationships among Nash equilibria, variational equilibria and dynamic equilibria for non-cooperative games. Cubiotti ([10]) and Cubiotti and Yen ([11]) prove the existence of a solution for gener-

alized quasi-variational inequalities in infinite-dimensional normed spaces under some conditions.

Relaxation algorithms are powerful methods for computing Nash equilibrium policies when the problem is not tractable enough to solve the necessary conditions of optimality. These iterative algorithms rely on averaging the current solution iterate with the solution of the best response problem each player solves. Uryas'ev and Rubinstein ([24]) study the convergence of such algorithms in finite dimensions. They find the equilibria of a non-cooperative game for some payoff function on a closed, compact space.

1.2.3 Closed-loop policy and reference price

A closed-loop policy usually performs better than an open-loop policy when solving problems with data uncertainty. Especially in the area of robust optimization, which is often criticized for being overly-conservative, researchers are interested in bringing the ideas of closed-loop policies or feedback control into this area in order to improve its performance. In [4], Ben-Tal et al. introduce an affinely adjustable robust counterpart (AARC) to model linear programs with uncertain parameters. They introduce the notion of adjustable variables in order to refer to those variables that can be chosen after the realization of the uncertain parameters. They show that AARC is significantly less conservative than the usual robust counterpart and is tractable in most cases. However, the demand is assumed to be exogenous in order to simplify the problem. This is not realistic in the pricing problem. Researchers also look at stochastic optimization to make this method more realistic and computationally tractable. Chen and others ([9]) introduce a unified framework of approximating multi-period stochastic optimization with safeguarding constraints, using ideas of robust optimization. They have shown that the framework is computationally tractable in the form of second order cone programming and scalable across periods.

Traditional economic marketing and operational models view the customers as rational agents who make decisions based on the current prices and market conditions. However, in a market with repeated interactions, customers' purchase decisions are

also determined by past observed prices. In [19], Popescu and Wu study the dynamic pricing problem of a monopolist firm, where demand is sensitive to the firm's past pricing history. The consumers form a reference price based on the pricing history, and their purchasing decisions are made by assessing prices as discounts or surcharges relative to the reference price. The model considered in this paper is a deterministic model, where all the parameters are known to be some nominal values.

1.3 The pricing problem and outline of the thesis

In this thesis, we consider a monopoly as well as an oligopoly pricing problem. In the latter, several sellers simultaneously seek a pricing policy for their products. The products are usually assumed to be differentiated and substitutable. Each seller has the option to set prices for her products at each time period, and her goal is to find a pricing policy that will yield the maximum overall profit. Each seller has a fixed initial inventory of each product to be allocated over the entire time horizon and does not have the option to produce additional inventory between periods. There are no holding or back-order costs. In addition, the products are perishable and have no salvage costs. This means that at the end of the entire time horizon, any remaining products will be worthless. The demand function is a linear function of the prices, and the parameters in the function are uncertain.

In order to address the demand uncertainty, some researchers have recently adopted ideas from robust optimization to solve the pricing problem in which the demand is only known to be within an uncertainty set rather than some nominal value. In order to simplify the problem, most papers such as [4] have assumed that the demand uncertainty is given exogenously and will not be affected by the seller's price. However, in practice, the demand for a product will usually be affected by the seller's price and her competitors' prices. In this thesis, we address demand uncertainty by modeling it as a function of all sellers' prices for the current period, and the past pricing history. We use the notation of reference price to model the past pricing history.

The main contributions of this thesis are as follows:

1. We describe various dynamic pricing models with demand uncertainty. We use ideas of robust optimization, adjustable robust optimization, dynamic programming and stochastic optimization to find adaptable closed-loop pricing policies. We study the tractability of solving these problems.

2. We study how the budget of uncertainty, the presence of a reference price, delayed resource allocation, and feedback control affect the quality of the pricing policies.

3. We address computational issues of these models in a multi-period setting.

4. We prove existence of solution for the best response subproblem as well as of the market equilibrium problem for all of the models we discuss in this thesis.

5. We discuss an iterative learning algorithm and illustrate computationally its convergence to an equilibrium pricing policy.

The thesis is outlined as follows. In Chapter 2, we introduce the notations that are used throughout the thesis. We formulate a number of models for the pricing problem. In Chapter 3, we consider a simplified setting, where there is a single seller and two time periods in the model. We compare different models in this setting and provide both theoretical and computational results regarding the effect of budget of uncertainty, the presence of a reference price, delayed resource allocation and the feedback control on the quality of the pricing policy. In Chapter 4, we extend the problem to a monopoly and multi-period setting. We discuss how to address the computational issues with the increase of time periods. In Chapter 5, we consider an oligopoly market where the competition among sellers affects each seller's decision. We study the existence of Nash equilibrium of these models. In Chapter 6, we conclude the thesis and discuss some possible future work in this research area.

Chapter 2

Model formulation

In this chapter, we introduce the notations and assumptions that will be used throughout the thesis. We discuss how to address the issue of uncertainty in demand and what we mean by robust policies. Each seller faces the problem of finding a pricing policy that maximizes her total profit from the sale of her inventory over the entire time horizon. We call this pricing problem that each seller faces the best response problem and the resulting pricing policy the best response policy. We focus on formulating the best response problem for each single seller and leave the competition issue aside in this chapter. Both the robust demand and stochastic demand are considered, and the robust optimization, stochastic optimization and dynamic programming are used to formulate the best response problem.

2.1 Notations

We denote the set of all sellers by $\mathbf{I}$ and a single seller by $i \in \mathbf{I}$. The set of all competitors of seller i is denoted by -i. The time horizon is divided into a finite number of time periods. We denote the ordered set of all time periods by $\mathbf{T}$ and a single time period by $t \in \mathbf{T}$.

At the beginning of the time horizon, each seller i starts with a given inventory of the products, denoted by C_i, and competes with her competitors by setting her price p_i^t at each period t. We denote the prices set by the competitors of seller i at period

t by p^t_{-i}. Seller i's pricing policy over the entire time horizon consists of the prices $(p_i^1, p_i^2, ..., p_i^T)$ and is denoted by $\mathbf{p}_i$. The pricing policy for all the sellers consists of the price vectors $(\mathbf{p}_1, \mathbf{p}_2, ...\mathbf{p}_I)$ and is denoted by $\mathbf{p}$. The demand function faced by seller i, at period t, is denoted by h_i^t.

2.2 Demand model

The demand function can assume various forms, depending on the assumptions we make on the model. In a monopoly market, if the seller's past pricing history does not have an impact on her demand function at the current period, the demand function is simply a function of her price at the current period, i.e., $h_i^t(p_i^t)$. In contrast, when the past pricing history is assumed to affect the current period's demand, the demand function is of the form $h_i^t(p_i^t, r_i^t)$, where r_i^t is the reference price of seller i at period t. The reference price r_i^t could be just the previous period's price p_i^{t-1} or an aggregated value of all the past prices, depending on how long we assume that the past pricing history can affect the future demand. Similarly, in a duopoly market, seller i's demand function will also be affected by her competitors' prices.

2.2.1 Demand function with uncertainty

In this thesis, we adopt a common demand model where the demand is a linear function of the price. For example, when we consider a monopoly setting and neglect the effect of the reference price, the demand of seller i at time t can be written as:

$$h_i^t = D_i^t - \alpha_i^t p_i^t$$

In practice, it is usually very difficult to determine the exact values of the parameters D_i^t and α_i^t. In this thesis, we assume that each of these parameters falls into allowed ranges with some known nominal values. For instance, we denote the nominal value of D_i^t by $\overline{D}_i^t$ and the deviation by $\sigma_{D_i^t}$. The realization of D_i^t belongs to an interval centered around $\overline{D}_i^t$ with half-length $\sigma_{D_i^t}$, i.e., $D_i^t \in [\overline{D}_i^t - \sigma_{D_i^t}, \overline{D}_i^t + \sigma_{D_i^t}]$.

We use ξ_i^t to denote the uncertainty factor in seller i's demand at period t.

2.2.2 Various demand functions

Assumption 1. When the reference effect is not considered, the demand function faced by seller i, at time period t, is only dependent on the prices set by all the sellers at period t. That is,

$$h_i^t(p_i^t, p_{-i}^t, \xi_i^t) = D_i^t - \alpha_i^t p_i^t + \beta_i^t p_{-i}^t, \tag{2.1}$$

where $\xi_i^t = (D_i^t, \alpha_i^t, \beta_i^t)$ denotes the uncertainty parameters and take any value in an uncertainty set U_i^t.

Assumption 2. When the reference effect is considered, the demand function faced by seller i, at period t, is dependent on the prices set by all the sellers at period t and the past pricing history. The latter is represented by a reference price that is denoted by r_i^t. In this case, the demand function is defined as:

$$h_i^t(p_i^t, p_{-i}^t, \xi_i^t, r_i^t) = D_i^t - \alpha_i^t p_i^t + \beta_i^t p_{-i}^t + \gamma_i^t r_i^t, \tag{2.2}$$

where $\xi_i^t = (D_i^t, \alpha_i^t, \beta_i^t, \gamma_i^t)$ denotes the uncertainty parameters and takes any value in an uncertainty set U_i^t.

Assumption 3. The demand function of seller i is a strictly decreasing linear function of her price, and an increasing linear function of her competitors' prices and her reference price. Mathematically, $D_i^t, \alpha_i^t, \beta_i^t, \gamma_i^t$ are strictly positive real values.

Assumption 4. We model the reference price as $r_i^t = g_i(p_i^{t-1}, r_i^{t-1})$, where g_i is a linear function of p_i^{t-1} and r_i^{t-1}. Furthermore, g_i increases with respect to p_i^{t-1} and with respect to r_i^{t-1}.

2.2.3 Budget of uncertainty

One problem with the robust optimization paradigm is that it may lead to an overly conservative solution that would allow uncertain parameters to be at the value corresponding to the worst-case scenario at all times. Nevertheless, such a scenario is highly unlikely to occur. To overcome this drawback of robust optimization, literature often introduces a budget of uncertainty Γ_i^t to bound the cumulative dispersion of the realized parameters around the nominal values over time (see [7]).

The budget of uncertainty is a very effective way to measure the trade-off between performance and conservativeness. A small budget of uncertainty gives less protection against data perturbation, but it gives better objective values. In contrast, a high budget of uncertainty gives better protection, but at the cost of performance.

The budget of uncertainty is chosen by seller i to reflect his attitude towards uncertainty and is data to the problem. When Γ_i^t is zero, seller i is considering a deterministic pricing problem, where all the uncertainty parameters are forced to take their nominal values. On the other hand, when Γ_i^t is large, seller i allows more variation or uncertainty on the parameters. In this case, seller i is considering the worst-case scenario, and the robust formulation becomes more conservative (i.e., risk-averse).

With the notation of budget of uncertainty, we can define the uncertainty set as follows:

- when the reference price effect is not considered, the uncertainty set U_i^t is defined as:

$$U_i^t = \left\{ \xi_i^t = (D_i^t, \alpha_i^t, \beta_i^t) : \left| \frac{D_i^t - \overline{D}_i^t}{\sigma_{D_i^t}} \right| + \left| \frac{\alpha_i^t - \overline{\alpha}_i^t}{\sigma_{\alpha_i^t}} \right| + \left| \frac{\beta_i^t - \overline{\beta}_i^t}{\sigma_{\beta_i^t}} \right| \leq \Gamma_i^t \right\},$$

  where $\overline{D}_i^t$, $\overline{\alpha}_i^t$, and $\overline{\beta}_i^t$ are the nominal values of the uncertainty parameters, $\sigma_{D_i^t}$, $\sigma_{\alpha_i^t}$, and $\sigma_{\beta_i^t}$ are the deviations of the uncertainty parameters around the nominal values, and Γ_i^t is the budget of uncertainty. All these parameters are data to the problem.

- when the reference price effect is considered, U_i^t becomes:

$$U_i^t = \left\{ \xi_i^t = (D_i^t, \alpha_i^t, \beta_i^t, \gamma_i^t) : \left| \frac{D_i^t - \overline{D}_i^t}{\sigma_{D_i^t}} \right| + \left| \frac{\alpha_i^t - \overline{\alpha}_i^t}{\sigma_{\alpha_i^t}} \right| + \left| \frac{\beta_i^t - \overline{\beta}_i^t}{\sigma_{\beta_i^t}} \right| + \left| \frac{\gamma_i^t - \overline{\gamma}_i^t}{\sigma_{\gamma_i^t}} \right| \leq \Gamma_i^t \right\}.$$

2.3 Best response problem

In this section, we formulate the best response problems for both the robust demand model and the stochastic demand model. In the robust demand model, the parameters in the demand function are uncertain and only known to belong to some uncertainty set. For this demand model, we have formulated four different robust optimization models, which are labeled as Robust-M1, Robust-M2, Robust-M3 and Robust-M3-AARC respectively. In these four robust optimization models, each seller adopts a pricing policy that is robust to all the possible realizations of the uncertain parameters. However, in the stochastic demand model, the parameters are assumed to follow a known probability distribution, and each seller adopts a policy that maximizes her expected total profit.

2.3.1 Robust demand model

Robust-M1

In this robust formulation, we do not consider the reference effect in the demand function. Furthermore, the decisions that each seller needs to make are both the price and amount of resource allocated for sale at each period. Let $\xi_i^t = (D_i^t, \alpha_i^t, \beta_i^t)$. The mathematical formulation is as follows:

$$\max_{p_i^t} \sum_{t=1}^{T} p_i^t \times \min_{\xi_i^t \in U_i^t} (D_i^t - \alpha_i^t p_i^t + \beta_i^t p_{-i}^t)$$

$$\text{s.t.} \sum_{t=1}^{T} \min_{\xi_i^t \in U_i^t} (D_i^t - \alpha_i^t p_i^t + \beta_i^t p_{-i}^t) \leq C_i,$$

$$p_i^t \geq 0, t = 1, 2, ..., T$$

where $U_i^t = \left\{ \xi_i^t = (D_i^t, \alpha_i^t, \beta_i^t) : \left| \frac{D_i^t - \overline{D}_i^t}{\sigma_{D_i^t}} \right| + \left| \frac{\alpha_i^t - \overline{\alpha}_i^t}{\sigma_{\alpha_i^t}} \right| + \left| \frac{\beta_i^t - \overline{\beta}_i^t}{\sigma_{\beta_i^t}} \right| \leq \Gamma_i^t \right\}, t = 1, 2, ..., T.$

As can be seen in the model above, the amount of resource allocated for sale at period t has been implicitly fixed to be $\min_{\xi_i^t \in U_i^t} (D_i^t - \alpha_i^t p_i^t + \beta_i^t p_{-i}^t)$. This means that the demand is no longer determined by nature (i.e., by the actual realization of uncertain parameters), but by the value of parameters corresponding to the worst-case scenario within the uncertainty set. In other words, we price considering the worst possible scenario of demand that might occur. The strength of this formulation is that it reduces the side effect caused by uncertainty, as the amount of resource allocated for sale at each period is decided at time zero and it is guaranteed to be sold out. Therefore, each seller can optimize and determine her profit at time zero, without worrying that her prices may cause a violation of the resource capacity constraint. Nevertheless, the drawback of this model is that the objective value is fixed and will not increase even when the actual realization of parameters are not corresponding to the worst-case scenario. Next, we will compare this model with other robust optimization models.

Robust-M2

This robust optimization model is the same as Robust-M1 except that we consider the reference price effect in the demand function. The mathematical formulation is as follows:

$$\max_{p_i^t} \sum_{t=1}^{T} p_i^t \times \min_{\xi_i^t \in U_i^t} (D_i^t - \alpha_i^t p_i^t + \beta_i^t p_{-i}^t + \gamma_i^t r_i^t)$$

$$\text{s.t.} \sum_{t=1}^{T} \min_{\xi_i^t \in U_i^t} (D_i^t - \alpha_i^t p_i^t + \beta_i^t p_{-i}^t + \gamma_i^t r_i^t) \leq C_i,$$

$$p_i^t \geq 0, t = 1, 2..., T$$

where $U_i^t = \left\{ \xi_i^t = (D_i^t, \alpha_i^t, \beta_i^t, \gamma_i^t) : \left| \frac{D_i^t - \overline{D}_i^t}{\sigma_{D_i^t}} \right| + \left| \frac{\alpha_i^t - \overline{\alpha}_i^t}{\sigma_{\alpha_i^t}} \right| + \left| \frac{\beta_i^t - \overline{\beta}_i^t}{\sigma_{\beta_i^t}} \right| + \left| \frac{\gamma_i^t - \overline{\gamma}_i^t}{\sigma_{\gamma_i^t}} \right| \leq \Gamma_i^t \right\}, t = 1, 2, ..., T$, and $r_i^1 = 0$.

Robust-M3

This robust optimization model differs from Robust-M1 in the sense that it allows nature decide the demand rather than pre-allocating the amount of resource for sale. Each seller only needs to decide her price set at each period. However, the prices have to be chosen such that the cumulative demand over the time horizon does not go beyond the total capacity the seller has for all the possible realizations of uncertain parameters. The reference price effect is not considered in this model. In addition, we cannot set $D_i^t - \alpha_i^t p_i^t + \beta_i^t p_{-i}^t \geq 0$ as a constraint for all the possible values in the uncertainty set in order to ensure the non-negativity of the demand, as this may make the price overly constrained and solution sub-optimal or even infeasible. In fact, the demand is only required to be non-negative for the realization of uncertain parameters that corresponds to the optimal solution. To address this issue, we use $\max\{p_i^t(D_i^t - \alpha_i^t p_i^t + \beta_i^t p_{-i}^t), 0\}$ to represent seller i's demand at period t in our model. In this case, seller i can set any price for her product. If the price chosen leads to a negative demand, a zero demand (rather than negative demand) is used in the objective function and the resource capacity constraint. By doing this, an optimal solution to the model can be found, as the seller is given freedom to set her prices, and the "correct" amount of demand is always used in the model. The mathematical formulation is as follows:

$$\max_{z, p_i^t} z$$

$$\text{s.t. } z \leq \sum_{t=1}^{T} \max\{p_i^t(D_i^t - \alpha_i^t p_i^t + \beta_i^t p_{-i}^t), 0\}, \forall (D_i^t, \alpha_i^t, \beta_i^t) \in U_i^t$$

$$\sum_{t=1}^{T} \max\{D_i^t - \alpha_i^t p_i^t + \beta_i^t p_{-i}^t, 0\} \leq C_i, \forall (D_i^t, \alpha_i^t, \beta_i^t) \in U_i^t$$

$$p_i^t \geq 0, t = 1, 2, ..., T$$

where $U_i^t = \left\{ \xi_i^t = (D_i^t, \alpha_i^t, \beta_i^t) : \left| \frac{D_i^t - \overline{D_i^t}}{\sigma_{D_i^t}} \right| + \left| \frac{\alpha_i^t - \overline{\alpha_i^t}}{\sigma_{\alpha_i^t}} \right| + \left| \frac{\beta_i^t - \overline{\beta_i^t}}{\sigma_{\beta_i^t}} \right| \leq \Gamma_i^t \right\}, t = 1, 2, ..., T.$

Robust-M3-AARC

This model is a modification of Robust-M3 by adding some feedback control into the model. When the feedback control is considered, seller i's decision for her price at period t is postponed to the end of period t-1. This is achieved by modeling p_i^t as an affine function of the uncertainty parameters $D_i^k, \alpha_i^k, \beta_i^k$ realized by the end of period t-1. That is,

$$p_i^t = \pi_i^t + \sum_{k=0}^{t-1} \theta_i^k D_i^k + \sum_{k=1}^{t-1} \omega_i^k \alpha_i^k + \sum_{k=1}^{t-1} \lambda_i^k \beta_i^k. \tag{2.3}$$

This idea is inspired by the Affinely Adjustable Robust Counterpart (AARC) introduced in [4], where the robust formulation with the feedback control has been proven to be less conservative than the traditional robust formulation. The mathematical formulation of this model is as follows:

$$\max_{z,\theta_i^k,\omega_i^k,\lambda_i^k} z$$

$$\text{s.t. } z \leq \sum_{t=1}^{T} \max\{p_i^t(D_i^t - \alpha_i^t p_i^t + \beta_i^t p_{-i}^t), 0\}, \forall (D_i^t, \alpha_i^t, \beta_i^t) \in U_i^t$$

$$\sum_{t=1}^{T} \max\{D_i^t - \alpha_i^t p_i^t + \beta_i^t p_{-i}^t, 0\} \leq C_i, \forall (D_i^t, \alpha_i^t, \beta_i^t) \in U_i^t$$

$$p_i^t = \pi_i^t + \sum_{k=0}^{t-1} \theta_i^k D_i^k + \sum_{k=1}^{t-1} \omega_i^k \alpha_i^k + \sum_{k=1}^{t-1} \lambda_i^k \beta_i^k$$

where $U_i^t = \left\{ \xi_i^t = (D_i^t, \alpha_i^t, \beta_i^t) : \left| \frac{D_i^t - \overline{D}_i^t}{\sigma_{D_i^t}} \right| + \left| \frac{\alpha_i^t - \overline{\alpha}_i^t}{\sigma_{\alpha_i^t}} \right| + \left| \frac{\beta_i^t - \overline{\beta}_i^t}{\sigma_{\beta_i^t}} \right| \leq \Gamma_i^t \right\}, t = 1, 2...T$, and $D_i^0 = 0, \alpha_i^0 = 0, \beta_i^0 = 0$.

2.3.2 Stochastic demand model

In this section, we consider a stochastic demand model, where the uncertain parameters are assumed to follow some known probability distribution. In the computational part of this thesis, we assume the uncertain parameters follow a uniform distribution.

Dynamic programming formulation

In the dynamic programming formulation, the state variable is the remaining inventory at the start of each period, which is denoted by s_i^t. As the uncertain parameters are assumed to follow a uniform distribution, we discretize them into a finite number of values with equal probabilities. Let $f_i^t(s_i^t)$ denote the optimal expected profit seller i can make from period t until the end of the time horizon with remaining inventory s_i^t. The reference effect is not considered in the demand function. The mathematical formulation is as follows:

$$f_i^t(s_i^t) = \max_{p_i^t} \mathbf{E}\{p_i^t \cdot \min(s_i^t, D_i^t - \alpha_i^t p_i^t + \beta_i^t p_{-i}^t) + f_i^{t+1}(s_i^{t+1})\}$$

$$s_i^{t+1} = s_i^t - \min(s_i^t, D_i^t - \alpha_i^t p_i^t + \beta_i^t p_{-i}^t)$$

$$f_i^{T+1}(s_i^{T+1}) = 0$$

Stochastic optimization formulation

In the stochastic optimization formulation, as before, we discretize the uncertain parameters into a finite number of values with equal probabilities. Each combination of these parameter values corresponds to a single scenario. Let ω index these scenarios, and let N denote the total number of scenarios we consider. Obviously, each scenario has a probability of $\frac{1}{N}$ to occur. The mathematical formulation is as follows:

$$\max \sum_{\omega=1}^{N} \sum_{t=1}^{T} \frac{p_{i,\omega}^t (D_{i,\omega}^t - \alpha_{i,\omega}^t p_{i,\omega}^t + \beta_{i,\omega}^t p_{-i,\omega}^t)}{N}$$

$$\text{s.t.} \sum_{t=1}^{T} (D_{i,\omega}^t - \alpha_{i,\omega}^t p_{i,\omega}^t + \beta_{i,\omega}^t p_{-i,\omega}^t) \leq C_i, \omega = 1, 2, ...N$$

$$p_{i,\omega}^t \geq 0, \omega = 1, 2, ..., N \text{ and } t = 1, 2, ..., T$$

2.4 Analysis of the robust best response problem

In this section, we study the existence of a solution to the robust optimization models that we have formulated in the previous section for a given competitor's price

$\mathbf{p}_{-i}$. We start by listing some assumptions that are assumed on the problem. These assumptions will be used to establish the existence of the solution.

2.4.1 Assumptions

Assumption 5. The price at period t for seller i is only allowed to vary between a minimum, denoted by $p^t_{i,min}$, and a maximum allowable level denoted by $p^t_{i,max}$. We require $p^t_{i,min}$ to be strictly positive and $p^t_{i,max}$ to be a level at which the demand for seller i vanishes irrespective of her competitors' prices at that period. Mathematically, we require that $p^t_{i,min} > 0$ and $\sup_{p^t_{-i},\xi^t_i} (h^t_i(p^t_{i,max}, p^t_{-i}, \xi^t_i)) = 0$ for all $t \in \mathbf{T}$.

Assumption 6. At any period t, for any fixed p^t_{-i} and $\xi^t_i \in U^t_i$, seller i's demand function $h^t_i(p^t_i, p^t_{-i}, \xi^t_i)$ is decreasing with respect to p^t_i over the set of feasible prices.

Assumption 5 ensures that the space of allowed prices is bounded, which is achieved by constraining the prices between some allowable upper and lower limits. With this condition, we eliminate strategies involving infinitely high price levels.

Assumption 6 ensures that the demand at any period for any seller does not increase with an increase in her price. In our demand model, as the demand is a linear function of prices, this condition is automatically satisfied.

2.4.2 Existence of solution

Proposition 1. Consider a fixed price for competitors, i.e., $\mathbf{p}_{-i}$. Then there exists a solution to the best response robust optimization model.

Proof. For Robust-M1 and Robust-M2, it is easy to show that the feasible space is non-empty and compact under Assumption 5. Since the demand function is a linear function of the price, the objective function is in fact a quadratic function of the price, p^t_i. This function is continuous and concave. Under these conditions, there exists a solution according to Weierstrass theorem (see [3]).

For Robust-M3 and Robust-M3-AARC, the presence of $\max\{p_i^t(D_i^t - \alpha_i^t p_i^t + \beta_i^t p_{-i}^t), 0\}$ makes the robust formulations non-convex. However, we can eliminate this problem by introducing a Mixed Integer Problem (MIP) formulation to replace $\max\{p_i^t(D_i^t - \alpha_i^t p_i^t + \beta_i^t p_{-i}^t), 0\}$. Details about this MIP formulation are shown in the next chapter. The outcome of this MIP formulation is that the resulting models of Robust-M3 and Robust-M3-AARC are still convex formulations, if we neglect the integral requirement on variable κ, which is used in the MIP formulation and can only take a value of either 0 or 1. One value of κ is required for each time period. Therefore, for an n-period pricing problem, there are a total of 2^n possible ways of assigning values to κ. For each of the value assignments of κ, the resulting models get rid of κ and become similar to Robust-M1 and Robust-M2, for which we have proven the existence of a solution. Therefore, to find the solution to Robust-M3 and Robust-M3-AARC, we just try all the 2^n possible ways of assigning values to κ, and the one that gives the best result will be a solution to these models. This shows that a solution exists for Robust-M3 and Robust-M3-AARC as well. □

Chapter 3

Uncertain data in a monopoly and two-period setting

We first consider a simplified pricing problem, where there is only one seller and two time periods in the model. The reason for studying this setting is that a single-seller, two-period problem is relatively easier to formulate and study, and yet, gives us insight into the pricing problem in a more general equilibrium setting. We discuss the approaches to solving all the best response models, which are the foundation of the equilibrium problem, presented in the previous chapter in this setting.

In this chapter, we study the six models we have formulated in the previous chapter, which are Robust-M1, Robust-M2, Robust-M3, Robust-M3-AARC, dynamic programming and stochastic optimization models. We provide both theoretical insights and computational results to show the effects of budget of uncertainty, reference price, delayed resource allocation and feedback control on the quality of the pricing policy.

3.1 Simplified notations

Since we only consider one seller and two periods in this chapter, to simplify the notations, we drop the seller's index i. Now, let p_t denote the seller's price at period t, where $t \in \{1, 2\}$. If the reference effect is not considered, the seller's demand at period t is defined as $h_t(p_t, \xi_t) = D_t - \alpha_t p_t$, where $\xi_t = (D_t, \alpha_t)$ denotes the uncertainty

parameters and takes any value in the uncertainty set U_t. Set U_t is defined as:

$$U_t = \left\{ \xi_t = (D_t, \alpha_t) : \left| \frac{D_t - \overline{D}_t}{\sigma_{D_t}} \right| + \left| \frac{\alpha_t - \overline{\alpha}_t}{\sigma_{\alpha_t}} \right| \leq \Gamma_t \right\}.$$

Similarly, if the reference effect is considered, the demand function at the second period is affected by the price at the first period, and it is defined as $h_2(p_1, p_2, \xi_2) = D_2 - \alpha_2 p_2 + \beta_2 p_1$, where $\xi_2 = (D_2, \alpha_2, \beta_2)$ can take any value in the uncertainty set U_2. Set U_2 is defined as:

$$U_2 = \left\{ \xi_2 = (D_2, \alpha_2, \beta_2) : \left| \frac{D_2 - \overline{D}_2}{\sigma_{D_2}} \right| + \left| \frac{\alpha_2 - \overline{\alpha}_2}{\sigma_{\alpha_2}} \right| + \left| \frac{\beta_2 - \overline{\beta}_2}{\sigma_{\beta_2}} \right| \leq \Gamma_2 \right\}.$$

Note that U_1 remains unchanged, as there is no past pricing history available at the start of the first period.

3.2 Solving the six monopoly models

In this section, we discuss the solution of the six models we considered in the previous chapter. In particular, we show how to transform the robust optimization models to their corresponding robust counterpart problems in order to solve these models efficiently. The dynamic programming and stochastic optimization models are solved through discretizing the parameters.

3.2.1 Robust optimization models

In the robust formulation, the capacity constraint has to be enforced for every possible realization of the uncertain parameters in the uncertainty set specified by the budget of uncertainty. Clearly, there is an infinite number of possible realizations of the parameters. This leads to an infinite number of constraints in the model. Clearly, we need to reformulate the original robust optimization model to a new problem with a finite number of constraints. The resulting robust representation is called the robust counterpart to the original robust optimization model.

Proposition 2. The minimum of the uncertain demand over the uncertainty set, denoted by $\min_{\xi_t \in U_t}(D_t - \alpha_t p_t)$, where $U_t = \left\{\xi_t = (D_t, \alpha_t) : \left|\frac{D_t - \overline{D}_t}{\sigma_{D_t}}\right| + \left|\frac{\alpha_t - \overline{\alpha}_t}{\sigma_{\alpha_t}}\right| \leq \Gamma_t\right\}$, will only occur at two extreme points, which are $(\overline{D}_t - \sigma_{D_t} \times \Gamma_t, \overline{\alpha}_t)$ and $(\overline{D}_t, \overline{\alpha}_t + \sigma_{\alpha_t} \times \Gamma_t)$. When the reference effect is considered, the minimum of the uncertain demand, denoted by $\min_{\xi_2 \in U_2}(D_2 - \alpha_2 p_2 + \beta_2 p_1)$ will only occur at three extreme points, which are $(\overline{D}_2 - \sigma_{D_2} \times \Gamma_2, \overline{\alpha}_2, \overline{\beta}_2)$, $(\overline{D}_2, \overline{\alpha}_2 + \sigma_{\alpha_2} \times \Gamma_2, \overline{\beta}_2)$ and $(\overline{D}_2, \overline{\alpha}_2, \overline{\beta}_2 - \sigma_{\beta_2} \times \Gamma_2)$, where $U_2 = \left\{\xi_2 = (D_2, \alpha_2, \beta_2) : \left|\frac{D_2 - \overline{D}_2}{\sigma_{D_2}}\right| + \left|\frac{\alpha_2 - \overline{\alpha}_2}{\sigma_{\alpha_2}}\right| + \left|\frac{\beta_2 - \overline{\beta}_2}{\sigma_{\beta_2}}\right| \leq \Gamma_2\right\}$.

Proof. The proof follows due to the linearity of the demand function. □

Robust counterpart of Robust-M1

Let S_t denote the set of extreme points specified in Proposition 2. The robust counterpart of Robust-M1 in a monopoly and two-period setting becomes:

$$\max \sum_{t=1}^{2} p_t \times \min_{(D_t, \alpha_t) \in S_t} (D_t - \alpha_t p_t)$$

$$\text{s.t.} \sum_{t=1}^{2} \min_{(D_t, \alpha_t) \in S_t} (D_t - \alpha_t p_t) \leq C,$$

$$p_t \geq 0, t = 1, 2,$$

where $\left|\frac{D_t - \overline{D}_t}{\sigma_{D_t}}\right| + \left|\frac{\alpha_t - \overline{\alpha}_t}{\sigma_{\alpha_t}}\right| \leq \Gamma_t, t = 1, 2.$

The robust counterpart of Robust-M1 can be implemented using any optimization tool such as ILOG.

Robust counterpart of Robust-M2

Similarly, the robust counterpart of Robust-M2 becomes:

$$\max \sum_{t=1}^{2} p_t \times \min_{(D_t,\alpha_t)\in S_t} (D_t - \alpha_t p_t + \gamma_t r_t)$$

$$\text{s.t. } \sum_{t=1}^{2} \min_{(D_t,\alpha_t)\in S_t} (D_t - \alpha_t p_t + \gamma_t r_t) \leq C,$$

$$p_t \geq 0, t = 1, 2$$

where $\left|\frac{D_t-\overline{D}_t}{\sigma_{D_t}}\right| + \left|\frac{\alpha_t-\overline{\alpha}_t}{\sigma_{\alpha_t}}\right| + \left|\frac{\gamma_t-\overline{\gamma}_t}{\sigma_{\gamma_t}}\right| \leq \Gamma_t, t = 1, 2$, and $r_1 = 0$.

Robust counterpart of Robust-M3

There is a problem with the representation of $\max\{p_t(D_t - \alpha_t p_t), 0\}$ in Robust-M3: if we introduce a slack variable z_t to replace $\max\{p_t(D_t - \alpha_t p_t), 0\}$, where $z_t \geq p_t(D_t - \alpha_t p_t)$ and $z_t \geq 0$, this formulation becomes unbounded, as it is a maximization problem.

In order to solve this problem, we adopt ideas from mixed integer programming (MIP) to represent $\max\{p_t(D_t - \alpha_t p_t), 0\}$.

Proposition 3. The robust counterpart of Robust-M3 can be represented as:

$$\max z$$

$$\text{s.t. } z \leq \sum_{t=1}^{2} z_t$$

$$\sum_{t=1}^{2} d_t \leq C$$

$$p_t = \pi_t + \sum_{k=0}^{t-1} \theta_k D_k + \sum_{k=1}^{t-1} \omega_k \alpha_k$$

$$z_t \leq p_t(D_t - \alpha_t p_t) + M(1-\kappa), \forall (D_t, \alpha_t) \in S_t$$

$$z_t \leq M\kappa$$

$$D_t - \alpha_t p_t \leq M\kappa, \forall (D_t, \alpha_t) \in S_t$$

$$D_t - \alpha_t p_t + M(1-\kappa) \geq 0, \forall (D_t, \alpha_t) \in S_t$$

$$d_t \geq D_t - \alpha_t p_t, \forall (D_t, \alpha_t) \in S_t$$

$$d_t \geq 0, t = 0, 1$$

$$\kappa \in \{0, 1\},$$

where $\left|\frac{D_t - \overline{D}_t}{\sigma_{D_t}}\right| + \left|\frac{\alpha_t - \overline{\alpha}_t}{\sigma_{\alpha_t}}\right| \leq \Gamma_t, t = 1, 2.$

Proof. To prove this result, we need to illustrate that we can replace $\max\{p_t(D_t - \alpha_t p_t), 0\}$ by z_t with the following additional constraints:

$$z_t \leq p_t(D_t - \alpha_t p_t) + M(1-\kappa) \quad (1)$$

$$z_t \leq M\kappa \quad (2)$$

$$D_t - \alpha_t p_t \leq M\kappa \quad (3)$$

$$D_t - \alpha_t p_t + M(1-\kappa) \geq 0 \quad (4)$$

$$\kappa \in \{0, 1\} \quad (5)$$

where M denotes a very large number.

We examine the following three cases to ensure that this is a correct representation:

1. If $D_t - \alpha_t p_t > 0$, then $\kappa = 1$. Subsequently, it follows that $z_t \leq p_t(D_t - \alpha_t p_t)$ from constraint (1) and $z_t \leq M$ from constraint (2). Since M is a very large number, $z_t \leq M$ becomes redundant, and we have $z_t \leq p_t(D_t - \alpha_t p_t)$ as the final result.

2. If $D_t - \alpha_t p_t = 0$, then $\kappa = 0$ or 1. In either case, it follows that $z_t \leq 0$ from constraint (1) and (2).

3. If $D_t - \alpha_t p_t < 0$, then $\kappa = 0$ due to (4). Subsequently, it follows that $z_t \leq p_t(D_t - \alpha_t p_t) + M$ from constraint (1) and $z_t \leq 0$ from constraint (2). Since M is a very large number, $z_t \leq 0$ is a tighter constraint and becomes the final result.

As we are maximizing z in the model, z_t is forced to be equal to $\max\{p_t(D_t - \alpha_t p_t), 0\}$.

In contrast, we can safely replace $\{D_t - \alpha_t p_t, 0\}$ with a dummy variable d_t in the resource capacity constraint, as d_t is bounded by capacity C.

Based on the analysis above, this MIP representation is equivalent to Robust-M3. More importantly, Robust-M3 remains a convex optimization problem, if we relax the integral requirement on κ. □

Robust counterpart of Robust-M3-AARC

Robust-M3-AARC has the same problem as Robust-M3 in the representation of $\max\{p_t(D_t - \alpha_t p_t), 0\}$. We use the same MIP formulation to resolve this problem. In addition, as the price is modeled as an affine function of the uncertain parameters realized in the past periods, the objective function becomes a quadratic function of uncertain parameters. Therefore, the realized values of uncertain parameters that lead to the worst-case scenario may not only occur at the two extreme points as we explained before. Instead, they can be any of the four extreme points, or points that make the resource capacity constraint tight. In this thesis, we make an approximation by only considering the four extreme points for each uncertain parameter. We study how good this approximation is through simulation at a later chapter. Let S_t denote

the set of four extreme points, which are $(\overline{D}_t - \sigma_{D_t} \times \Gamma_t, \overline{\alpha}_t)$, $(\overline{D}_t + \sigma_{D_t} \times \Gamma_t, \overline{\alpha}_t)$, $(\overline{D}_t, \overline{\alpha}_t + \sigma_{\alpha_t} \times \Gamma_t)$, $(\overline{D}_t, \overline{\alpha}_t - \sigma_{\alpha_t} \times \Gamma_t)$. The robust counterpart becomes:

$$\max z$$

$$\text{s.t. } z \leq \sum_{t=1}^{2} z_t$$

$$\sum_{t=1}^{2} d_t \leq C$$

$$p_t = \pi_t + \sum_{k=0}^{t-1} \theta_k D_k + \sum_{k=1}^{t-1} \omega_k \alpha_k$$

$$z_t \leq p_t(D_t - \alpha_t p_t) + M(1-\kappa), \forall (D_t, \alpha_t) \in S_t$$

$$z_t \leq M\kappa$$

$$D_t - \alpha_t p_t \leq M\kappa, \forall (D_t, \alpha_t) \in S_t$$

$$D_t - \alpha_t p_t + M(1-\kappa) \geq 0, \forall (D_t, \alpha_t) \in S_t$$

$$d_t \geq D_t - \alpha_t p_t, \forall (D_t, \alpha_t) \in S_t$$

$$d_t \geq 0, t = 0, 1$$

$$\kappa \in \{0, 1\},$$

where $\left|\frac{D_t - \overline{D}_t}{\sigma_{D_t}}\right| + \left|\frac{\alpha_t - \overline{\alpha}_t}{\sigma_{\alpha_t}}\right| \leq \Gamma_t, t = 1, 2.$

3.2.2 A heuristic approach to solving the dynamic programming model

In the dynamic programming model, the state variable, denoted as s_t, is the inventory left at the start of period t. As the state variable can take any value between 0 and the total capacity C, it is an infinite-state dynamic programming problem. Our approach to handling the infinite-state problem is to discretize the state variable into a finite number of discrete values. When the discretization becomes fine, the dynamic programming model gives solutions equal or close to the optimal solution.

Similarly, we assume that the uncertain parameters follow a uniform distribution

and discretize them into a finite number of values. The dynamic programming model in a monopoly and two-period setting becomes:

$$f_t(s_t) = \max_{p_t} \mathbf{E}\{p_t \cdot \min(s_t, D_t - \alpha_t p_t) + f_{t+1}(s_{t+1})\}$$
$$s_{t+1} = s_t - \min(s_t, D_t - \alpha_t p_t)$$
$$f_{T+1}(s_{T+1}) = 0$$

In order to solve the dynamic programming model at each stage, we need to decide the optimal way of splitting the remaining inventory s_t between the resource used at the current period and the resource carried over to the next period. We denote the amount of inventory used at period t by s_u and the amount of inventory carried over to period $t+1$ by s_l, i.e., $s_t = s_u + s_l$. Clearly, s_u can take any value between 0 and s_t. To find the optimal value for s_u, we discretize s_t into a finite number of values. We try each of these values for s_u and solve the problem. By doing this, we try all the possible ways of splitting the inventory between the current stage and the next stage. The optimal solution corresponds to the split that gives the highest expected profit. Both D_t and α_t are discretized into N values. We index the values of (D_t, α_t) by k. There are N^2 ways of discretizing (D_t, α_t) in total. Each has a probability of $\frac{1}{N^2}$ to occur. We first ignore the resource capacity constraint and solve the dynamic programming model at each step as follows:

$$f_t(s_t) = \max_{p_t} \mathbf{E}\{p_t \cdot \min(s_t, D_t - \alpha_t p_t) + f_{t+1}(s_{t+1})\}$$
$$\Downarrow$$
$$f_t(s_t) = \max_{p_t} \sum_{k=1}^{N^2} p_t \cdot (D_t^k - \alpha_t^k p_t) \cdot \frac{1}{N^2} + f_{t+1}(s_{t+1})$$

Next, we choose p_t that makes the derivative of the current stage's profit equal to

zero.

$$\nabla_{p_t}(\sum_{k=1}^{N^2} p_t \cdot (D_t^k - \alpha_t^k p_t)) = \sum_{k=1}^{N^2} (D_t^k - 2\alpha_t^k p_t) = 0$$

$$\Downarrow$$

$$p_t = \frac{\sum_{k=1}^{N^2} D_t^k}{\sum_{k=1}^{N^2} 2\alpha_t^k}$$

Based on p_t derived above, the expected demand, denoted by $E(D)$, is computed using the nominal value of D and α:

$$E(D) = \overline{D}_t - \overline{\alpha}_t p_t$$

The optimal decision for the price at the current stage, denoted as p_t^{opt}, depends on the following two cases:

1. If $E(D) \leq s_u$, the resource allocated for the current stage is sufficient for setting p_t as the optimal price for the current stage. Therefore, $p_t^{opt} = p_t$, and $f_t(s_t) = p_t^{opt} \cdot E(D) + f_{t+1}(s_t - E(D))$.

2. If $E(D) \geq s_u$, the demand determined by p_t is greater than the amount of resource being allocated. As it is a quadratic function, the optimal price will make the determined demand equal to s_u, i.e., $p_t^{opt} = \frac{\overline{D}_t - s_u}{\overline{\alpha}_t}$, and $f_t(s_t) = p_t^{opt} \cdot s_u + f_{t+1}(s_t - s_u)$.

3.2.3 Stochastic optimization model

Stochastic optimization is another framework for modeling optimization problems that involve uncertainty. In the robust formulation, the parameters are assumed to be known only within bounds, and the goal is to find a solution which is feasible for all such data and optimal in some sense. Stochastic optimization models are similar in style but take advantage of the fact that probability distribution governing the data are known or can be estimated. The goal here is to find some policy that is feasible

for all (or almost all) the possible data instances and maximizes the expectation of some function of the decisions and the random variables.

We assume the uncertainty parameters follow a uniform distribution. For a two-period pricing problem, we discretize each of uncertain parameters into four values. Therefore, we have $(4 * 4)^2 = 256$ possible scenarios in total.

3.3 Comparison of six models

In this section, we examine all six models we presented and study how the effect of a reference price, budget of uncertainty, delayed resource allocation and feedback control affect the pricing policy.

3.3.1 Test instances

Five test instances, labeled from test1.dat through test5.dat, were designed for testing. All the input parameters were generated randomly, and the budget of uncertainty Γ_t was randomly chosen as long as both $\overline{D}_t - \Gamma_t \varepsilon_{D_t} \geq 0$, and $\overline{\alpha}_t - \Gamma_t \varepsilon_{\alpha_t} \geq 0$ hold.

3.3.2 The effect of reference price

Proposition 4. The presence of the reference price improves the objective value of the resulting pricing policy. In other words, the objective value of Robust-M2 is higher than that of Robust-M1.

Proof. Let z1 and z2 be the optimal objective values of Robust-M1 and Robust-M2, respectively, and let (p_1^*, p_2^*) be the optimal prices in Robust-M1. There are two possible cases:

1. If (p_1^*, p_2^*) satisfies the resource capacity constraint in Robust-M2, it is also a feasible solution to Robust-M2. Let $z1'$ be the objective value of Robust-M2 with solution (p_1^*, p_2^*). Clearly, $z1'$ is greater than $z1$, as $\min_{\xi_2 \in U_2} (D_2 - \alpha_2 p_2^* + \beta_2 p_1^*) \geq \min_{\xi_2 \in U_2} (D_2 - \alpha_2 p_2^*)$. Hence, $z2 \geq z1' \geq z1$.

2. If (p_1^*, p_2^*) violates the resource capacity constraint in Robust-M2, increase p_2^* to $p_2^* + \Delta p_2$ such that $\min_{\xi_1 \in U_1}(D_1 - \alpha_1 p_1^*) + \min_{\xi_2 \in U_2}(D_2 - \alpha_2(p_2^* + \Delta p_2) + \beta_2 p_1^*) = C$. Note that the price and demand at the first period remain the same, which implies that the profit obtained at the first period remains the same in both models. However, in Robust-M2, the demand at the second period is $\min_{\xi_2 \in U_2}(D_2 - \alpha_2(p_2^* + \Delta p_2) + \beta_2 p_1^*)$, which is equal to $C - \min_{\xi_1 \in U_1}(D_1 - \alpha_1 p_1^*)$, while in Robust-M1, the demand at the second period is $C - \min_{\xi_2 \in U_2}(D_2 - \alpha_2 p_2^*)$, which is less or equal to $C - \min_{\xi_1 \in U_1}(D_1 - \alpha_1 p_1^*)$. This shows that the demand at the second period in Robust-M2 is at least equal to that in Robust-M1. As a result, the profit obtained at the second period in Robust-M2 is greater than that in Robust-M1. Therefore, $z2 \geq z1$.

□

Computational results

As can be seen in Table A.1, the profit obtained in Robust-M2 is greater than that in Robust-M1 for all the five test instances.

3.3.3 The effect of the budget of uncertainty

Proposition 5. The increase in the budget of uncertainty will decrease the objective value in the resulting pricing policy.

Proof. The proof follows easily: for the same set of uncertain parameters, a higher budget of uncertainty leads to a larger uncertainty set in the sense that it is a superset of the uncertainty set corresponding to a smaller budget uncertainty. As we consider the worst-case profit with respect to the uncertain parameters within the uncertainty set, for fixed prices, the profit obtained with a larger budget of uncertainty will always be less or equal to that obtained with a smaller budget of uncertainty. □

However, the robust optimization model is designed so that there is no violation of constraints for any realization of uncertain parameters that lie within the budget of

uncertainty. Since the actual realizations of uncertain parameters may not satisfy the budget of uncertainty, it is possible that the resource capacity constraint is violated by the robust solution. We expect that the higher the budget of uncertainty, the less likely that the robust solution violates the resource capacity constraint, since the protection level increases with the budget of uncertainty.

Computational results

In order to study how the budget of uncertainty affects the quality of the pricing policy, we increase the budget of uncertainty in test4.dat, which is 0.1 for both periods, using a step of 0.1 and obtain eight new test instances, which are labeled from test6.dat through test13.dat. We run Robust-M3 on all these nine test instances to see how the budget of uncertainty affects the profit obtained by the robust solution. We also assume the uncertain parameters follow a uniform distribution. We generate a large number of sample points of the actual realization of uncertain parameters and compute the probability of constraint violation for each budget of uncertainty.

The profits obtained by these nine test instances are shown in Table A.2, and the corresponding plot is shown in Figure B-1. We can see that with the increase of the budget of uncertainty from 0.1 to 0.9, the profit obtained decreases from 137.603 to 0.45. However, the probability of constraint violation also drops from 45.05% to 3.98%. This shows that an increase of budget of uncertainty has double effects on the robust solution. On the one hand, the profit obtained decreases, as the seller adopts a more conservative attitude towards uncertainty. On the other hand, the probability of constraint violation decreases too, as the protection level increases.

3.3.4 The effect of the delayed resource allocation

In this section, we compare Robust-M1 (this model determines the amount of resources for sale at time zero) with Robust-M3 and Robust-M3-AARC (these models let nature decide the demand based on the price set for that period and the actual realization of the uncertain parameters). We are interested in knowing which way of

determining the resource allocation leads to a better pricing policy, and what are the advantages and disadvantages with each type of resource allocation.

Computational results

The profits obtained by Robust-M1, Robust-M3 and Robust-M3-AARC for five test instances are shown in Table A.3. As can be seen in the table, the worst-case profits obtained by Robust-M1 are higher than those of Robust-M3 and Robust-M3-AARC for test2.dat and test5.dat, and the same for the rest of the three test instances. Another interesting observation is that the profits obtained by Robust-M3 and Robust-M3-AARC are the same for all the five test instances.

This result is not surprising. Although Robust-M1 determines the resource allocation at time zero (which may appear to be less flexible than Robust-M3 and Robust-M3-AARC), it actually gets rid of the risk of violating the resource capacity constraint by the realization of uncertain parameters. In Robust-M1, the resource allocated at each period is pre-fixed to be $\min_{(D_t,\alpha_t)\in S_t}(D_t - \alpha_t p_t)$. As set S_t only contains the two extreme points, we know exactly where the worst-case scenario can happen, and therefore, we can do a better optimization and achieve a higher profit. In contrast, although Robust-M3 and Robust-M3-AARC have the flexibility of letting nature decide the demand based on the actual realized value of parameters, they have to be better protected for whatever might occur in the future. In Robust-M3 and Robust-M3-AARC, the prices are chosen so that the sum of the highest possible demands is still less than the total capacity C. With this constraint, the prices have to be chosen relatively high, as otherwise, the sum of corresponding demands may exceed the total capacity. However, in the calculation of the objective value, the worst-case scenario is still considered in these two models. In contrast, as Robust-M1 pre-fixes the resource allocation to the worst-case demand, it allows the prices to be more flexible, and the sum of demand will not exceed the total capacity. Therefore, when we maximize the worst-case profit, Robust-M1 gives a better objective value than that of Robust-M3 and Robust-M3-AARC.

However, the advantages of Robust-M3 and Robust-M3-AARC lie in the flexibility

of the demand being determined by the actual realization of uncertain parameters, and this makes these two models work better on the actual realization of uncertain parameters. In contrast, as both the price and demand are pre-determined in Robust-M1, the profit obtained will not change with respect to the actual realization of uncertain parameters. In order to see the effect of the delayed resource allocation, we compare the total profits obtained by these three models on the first five test instances. A Matlab code was written to randomly generate a substantially large number of sample points of the actual realizations of uncertainty parameters, and compute the total profits obtained by these three models.

The comparison of the profits obtained by these three models for five test instances are shown from Figure B-2 to Figure B-6. From these figures, we can see that for test1.dat, test3.dat and test4.dat, since the worst-case profits obtained by these three models are the same, the profits obtained by Robust-M3 and Robust-M3-AARC for all the sample points are above or just cutting the profit obtained by Robust-M1. Even for test2.dat and test5.dat, although the worst-case profit of Robust-M1 is higher than those of the other two models, we can still see that for the majority of sampling points, the actual profits obtained by Robust-M3 and Robust-M3-AARC are higher than that of Robust-M1. These results show that although Robust-M3 and Robust-M3-AARC may "lose" to Robust-M1 in the worst-case scenario, they, in general, outperform Robust-M1.

We also count the percentage of sample points for which the profit of Robust-M1 is greater than that of Robust-M3 and Robust-M3-AARC, respectively. The result is shown in Table A.4. We can see that even for test2.dat, there are only 0.55% and 1.46% from the total sample points for which Robust-M1 outperforms Robust-M3 and Robust-M3-AARC, respectively. For test5.dat, the ratios are higher, but still, there are only 13.21% of sample points for which Robust-M1 has a higher profit than the other two models.

From these comparisons, we conclude that determining the resource allocation at time zero has the advantage of reducing the difficulty caused by the uncertain parameters, and therefore, improves the performance in the worst-case scenario. However,

the cost associated with this early resource allocation is that it loses the flexibility of adjusting to the actual realization of parameters. Hence, the actual performance of Robust-M1 is not as good as that of the other two models in general.

3.3.5 The effect of feedback control

Robust-M3 and Robust-M3-AARC are exactly the same, except that in Robust-M3-AARC, we incorporate the idea of feedback control by modeling the price at the second period as an affine function of the uncertainty realized at the first period. In this section, we compare Robust-M3 with Robust-M3-AARC to see how the feedback control affects the pricing policy.

We can see from Table A.3 that the worst-case profits obtained by these two models on all the five test instances are the same. This is as expected: if we look closely at the formulations of these two models, the only difference is that the second period's price is modeled as an affine function of D_1 and α_1 in Robust-M3-AARC. This should not cause any difference to the worst-case profits. For any optimal price to Robust-M3, we can always find coefficients for the affine function such that the second period's price in Robust-M3-AARC is the same as that in Robust-M3. Conversely, for any set of coefficients of the affine function in the optimal solution of Robust-M3-AARC, we can always set the second period's price in Robust-M3 to the value of that affine function in Robust-M1. Therefore, the profits obtained in the worst-case scenario should be the same for both models.

However, the affine function gives Robust-M3-AARC the ability to adjust its prices based on the uncertain parameters realized before. We can treat this as a tuning effect that adjusts the prices, most likely, towards an increase of the overall profit. The tuning effect of the affine function can be intuitively seen as follows: according to the affine function, p_2 increases if D_1 increases or α_1 decreases. When D_1 increases or α_1 decreases, it is usually a sign that the realized demand at period 1 is higher than what the seller has estimated. Therefore, considering the total limited amount of resources the seller has, the resources left for sale at period 2 might be less than she has expected to sell. The best response now is to increase the price at the second

period so that the demand at period 2 will decrease, and the profit obtained in period 2 can be maximized. We can see from Figure B-2 to Figure B-6 that the profit obtained by Robust-M3-AARC is, in general, above that of Robust-M3.

It is worth pointing out that the affine function does not guarantee the improvement in the objective value. Sometimes, it may even cause the profit to be lower than that obtained in Robust-M3 for some realization of uncertain parameters (see Table A.5). This is reasonable. The affine function may not be a good modeling of the relationship between the price and parameters realized before. A higher order function may be a better modeling tool; however, it will introduce much more complexity to the model. Nevertheless, we still expect that with the introduction of the affine function, Robust-M3-AARC will outperform Robust-M3 for most of the sample points. To see this, we count the percentage of sample points for which the profit of Robust-M3 is higher than that of Robust-M3-AARC, which is shown in Table A.5. We can see from the table that the highest percentage of such points is still less than 25%. This shows that the affine function does help to improve the pricing policy on average.

3.3.6 Comparison of the dynamic programming model with the robust optimization models

In this section, we compare the profits obtained by Robust-M1, Robust-M3-AARC and the dynamic programming model for all five test instances.

The comparison of the profits obtained by these three models for five test instances are shown from Figure B-7 to Figure B-11. We can see from these figures that the profits obtained by Robust-M3-AARC and dynamic programming are very close for most of the sample points, and both of these models have a higher profit than that of Robust-M1, in general. Taking a close look from Figures B-7 to Figure B-11, we observe that the curve obtained by the dynamic programming model is just slightly above that of Robust-M3-AARC. For some of sample points, Robust-M3-AARC even outperforms the dynamic programming model. The percentage of points for which the

dynamic programming "loses" to the other models may be caused by the discretization of uncertain parameters and the state variable we have performed in order to solve the dynamic programming model. As as result, this only provides an incomplete look-up table. A finer discretization will help to improve the performance of the dynamic programming model.

We count the number of sample points for which the profit obtained by Robust-M3-AARC is greater than that of the dynamic programming model, which is shown in Table A.6. We can see from this table that for most of the test instances, for more than 10% of the total sample points, Robust-M3-AARC has a higher profit than that of the dynamic programming model. Specially for test5.dat, the percentage of such points is as high as 28%. This comparison shows that although the dynamic programming model does give the best pricing policy among all these models, Robust-M3-AARC gives a result very close to that of the dynamic programming model or even outperforms it for some of sample points.

3.3.7 Comparison of the stochastic optimization model with the robust optimization models

In this section, we compare the profits obtained by Robust-M1, Robust-M3, Robust-M3-AARC and the stochastic optimization model.

The profits obtained by these models for five test instances are shown in Table A.7. As can be seen in the table, the stochastic optimization model gives a higher profit than the rest of the models for all the five test instances. This is because the stochastic optimization model is maximizing the expected profit, while the rest of models are maximizing the worst-case profit.

3.3.8 Evaluation of the four-point approximation in Robust-M3-AARC

We have mentioned above that we only consider four extreme points of the uncertain parameters to represent the worst-case scenario. As the objective function is a

quadratic function of uncertain parameters in Robust-M3-AARC, this approximation may not be sufficient to represent the worst-case scenario. In this section, we examine how good this approximation is through simulation.

We randomly generate a large number of sample points of (D_t, α_t) within the uncertainty set. We use the pricing policy returned by Robust-M3-AARC to compute the actual profit for each of the sample points we have generated. We compare the actual profit against the worst-case profit obtained by Robust-M3-AARC. If the actual profit is lower than the profit obtained by Robust-M3-AARC, we find a sample point that leads to a lower objective value than that obtained by the four extreme points we consider in the model. We compute the percentage of such sample points to see how effective the four-point approximation is in representing the worst-case scenario. In the simulation, we generated 10,000 sample points. The results obtained for all the five test instances are shown in Table A.8.

As we can see from the table, the percentages of such sample points are zero for test1.dat, test3.dat and test4.dat. For test2.dat and test5.dat, the percentages of such sample points are only 2.5% and 0.015%, respectively. Based on these results, we conclude that the four-point approximation is very effective in representing the worst-case scenario in Robust-M3-AARC.

Chapter 4

Uncertain data in a monopoly and multi-period setting

One of the challenges faced in the multi-period extension of the pricing problem is that the size of the problem grows exponentially with the number of periods. This problem applies not only to the dynamic programming and stochastic optimization models, but also to all the four robust optimization models we considered. In this chapter, we examine how to address the computational issues in the multi-period setting.

4.1 Dynamic programming model and stochastic optimization model

As discussed before, dynamic programming suffers badly from the "curse of dimensionality." This drawback makes dynamic programming inappropriate for solving large-scale problems. In our pricing problem, we have to discretize not only the state variable, but also the uncertain parameters in order to solve the problem at each stage. For instance, even if we just discretize the state variable into 100 intervals and the uncertain parameters into 10 intervals (this may not be big enough for an accurate estimation), the computational complexity will become of the order of $(100 * 10 * 10)^n$

for an n-period pricing problem. Clearly, the computational complexity explodes with the number of periods.

We conduct an experiment to record the running time required to solve the dynamic programming model for different time periods. We start with two periods and increase the number of time periods by 1 at each step. The results are shown in Table A.9. We can see that the running time explodes with the increase of the time period. From two periods to three periods, the running time increases by almost 200 times. When the number of periods is four, it takes more than 30 minutes to complete. This shows that although the dynamic programming model gives the best result, it is not easy to extend to the multi-period setting.

Similarly, although stochastic optimization does not suffer from the curse of dimensionality in the state variables, it is usually limited to problems with very few periods. When the number of periods increase, the running time explodes due to the exponential explosion of the event tree.

In conclusion, neither the dynamic programming model nor the stochastic optimization model extends easily to the multi-period setting.

4.2 Robust optimization models

The robust optimization modeling has been successfully applied to some large-scale and highly complex optimization problems. In our problem, as the demand is a linear function of the uncertain parameters, we identify that the value of uncertain parameters that leads to the worst-case scenario can only happen at the extreme points. As a result, the total number of constraints we need to consider for an n-period pricing problem is of the order of 2^n for the first three robust optimization models and 4^n for Robust-M3-AARC. In addition, as we only had access to the trial version of AMPL, we are limited to solve a problem with at most 300 constraints. However, for a eight-period pricing problem, the total number of constraints we have is about 900 constraints for the first three robust models, and more than 200,000 constraints for Robust-M3-AARC.

Thus, the main challenge faced in extending these models to a multi-period setting is how to handle the potentially exponential number of constraints as the number of periods grows. In particular, we have to first find a convenient way to generate the large number of constraints. Second, we need to find strategies to solve the problem, which is large-scale quadratic optimization problem with quadratic constraints, and is even non-convex for Robust-M3-AARC.

4.2.1 Approaches

A constraint generator is written to generate all the extreme points and write them to a file. This file can later be read by the main program to generate constraints as needed.

For the first three robust models, we use delayed constraint generation to solve the problems. By using this technique, we can solve a smaller problem at each time and yet, still be able to guarantee the optimal solution. For Robust-M3-AARC , as the formulation is not convex, we try two methods to solve the problem. The first method is to try different initial points and choose the best solution among all the solutions found as the optimal solution, assuming we have tried a substantially large number of initial points. The second method is to transform the non-convex problem to a convex problem with the MIP formulation, and then solve it with integer relaxation and branch-and-bound techniques.

4.2.2 Delayed constraint generation for solving Robust-M1 and Robust-M2

In the delayed constraint generation, we first choose a subset of the constraints that will be considered in the problem. The selected set of constraints are called active constraints and the set is called the active set, while the rest of the constraints are called inactive constraints. The smaller version of the original problem is solved to optimality and the obtained solution is then checked for feasibility among the inactive constraints. Any of the inactive constraints that are violated is added back to the

active set. The problem is then solved again for the new active set, and the obtained solution is checked against all the inactive constraints. This procedure repeats until a solution has been found to satisfy all the inactive constraints.

As the total number of constraints our solver can handle is limited to 300, we can not allow the size of the active set to grow without control. To maintain the size of the active set within a certain range, we remove any constraint which is not tight from the active set once some new constraint is added to the set.

4.2.3 Approaches to solving Robust-M3-AARC

Robust-M3-AARC is a non-linear, non-convex optimization problem with a huge number of constraints. A non-convex optimization problem cannot guaranteed to be solved to global optimality. Nevertheless, it is possible to find a close enough local optimum. To solve this non-convex problem, we tried two methods: the first method is to try different initial subsets of constraints and solve each of them using delay constraint generation individually; the best solution found among them is used as the optimal solution. The second method is to use the MIP formulation to transform the problem to a convex formulation, and solve it with integer relaxation and branch-and-bound techniques.

In the first method, we consider various starting points and solve the problem to optimality. A starting point refers to the active set in this case. It can be shown that each active set gives a different optimal value. A random subset generation algorithm is used to generate various starting points and identify the optimal solution from the given pool.

In the second method, we considered the MIP formulation of the original model. The new formulation is convex, if we relax the integrality constraints on the variable κ, and it can then be solved using AMPL. The solution obtained from this relaxed problem will become an upper bound of the original problem. A branch-and-bound technique is then applied to make each κ either 0 or 1 and find the optimal solution.

4.3 Computational results

In this section, we show that the conclusions we have drawn about the effect of the reference price, the budget of uncertainty, the delayed resource allocation, and the feedback control in the previous chapter still apply in a multi-period setting.

We randomly generated three test instances and labeled them as test8-1.dat, test8-2.dat and test8-3.dat, respectively.

The effect of the reference price

In this section, we compare the profits obtained by Robust-M1 and Robust-M2 for the 8-period test instances. The results are shown in Table A.10. We can see from the table that the profits obtained by Robust-M2 are higher than that of Robust-M1 for all the test instances.

The effect of the budget of uncertainty

In this section, we choose test8-1.dat, for which the budget of uncertainty is 0.1 for all the time periods, and increase the budget of uncertainty using a step of 0.1 to generate another eight test instances. The profits obtained by Robust-M1 on these nine test instances are shown in Figure B-12. Clearly, we see the profit obtained decreases when the budget of uncertainty increases.

The effect of the delayed resource allocation and feedback control

In this section, we compare the profits obtained by Robust-M1, Robust-M3 and Robust-M3-AARC on test8-1.dat to see whether the delayed resource allocation leads to a better pricing policy for most of the realizations of the uncertain parameters. Similar to what we have done in the 2-period case, we randomly generate a large set of sample points, where each of the sample points contains one set of realized values for the parameters, and compare the profits obtained by these three models on these sample points. The result is shown in Figure B-13. We can see from the figure that although the worst-case profit obtained by Robust-M1 is higher than that of the

other two models, it has a lower profit for most of the sample points. This shows that the delayed resource allocation has the strength of achieving higher profit even in a multi-period setting.

To see the effect of feedback control, we compare the results obtained by Robust-M3 and Robust-M3-AARC. We can see from Figure B-13 that the curve obtained by Robust-M3-AARC is slightly above that of Robust-M1. In fact, for 38% of the total sample points, Robust-M3 has a higher profit than Robust-M3-AARC.

Chapter 5

Uncertain data in an oligopoly setting

In this chapter, we consider the pricing problem in an oligopoly setting. We show that a market equilibrium pricing policy exists for all of our models. An iterative learning algorithm is studied for computing market equilibrium prices.

5.1 Existence of Nash equilibrium policy

Definition (Nash equilibrium policies). The pricing policies for each seller are Nash equilibrium pricing policies if no single seller can increase her payoff by unilaterally changing her policy.

Theorem 1. If an integer solution κ exists in the MIP best response formulations of Robust-M3 and Robust-M3-AARC, then there exists an equilibrium policy to these two models

Proof. If an integer solution κ exists in the MIP best response formulations of Robust-M3 and Robust-M3-AARC, we can eliminate κ from these two models. Thus, Robust-M3 and Robust-M3-AARC become similar to Robust-M1 and Robust-M2 in the sense that the feasible space is a non-empty, compact and convex set. As the objective

function is continuous and concave, there exists an equilibrium policy to these two models according to the Glicksburg-Fan-Debreu theorem (see [13]). □

Although a Nash equilibrium exists for all of our models, we may not be able to compute the equilibrium pricing policy for all of them as easily. In fact, only for Robust-M1, where both the price and demand are decided at time zero, is the equilibrium pricing policy computable, while for the rest of models, it may not. A detailed discussion follows:

Case 1: In Robust-M1, as the price and demand are decided at the start of the first period, the profit of each seller can be determined at the beginning of the game. In an oligopoly setting, the profit of each seller is only affected by the prices of her competitors, but not other factors such as the actual realization of the uncertain parameters. As in this robust model, the demand function is concave and continuous in all arguments, the joint strategy set is closed, convex, bounded, a Nash equilibrium can be computed easily(see [20]).

Case 2: For the rest of our formulations, either the price is determined at time zero and the actual realization of uncertain parameters determines the demand, or the decision for the prices are delayed until the realization of uncertain parameter at early periods are known. In both cases, the actual realization of uncertain parameters plays a crucial role in determining the profit that each seller can have. In other words, in an oligopoly setting, each seller's profit is no longer only affected by her own prices, but also by the actual realizations of the uncertain parameters in her model. In this case, a convergence to the equilibrium pricing policy may not be easily computed. One would need to compute an equilibrium pricing policy that is optimal for every possible realization of uncertain parameters. For different realizations of the uncertain parameters, the equilibrium pricing policy will change accordingly.

Although the equilibrium pricing policy may not be computable for some of our models, it is still likely to be achieved under certain assumptions. These are discussed in the following sections.

5.2 Iterative learning algorithms

In this section, we use the iterative learning algorithm as introduced in [17] to compute the market equilibrium prices. The basic idea for the algorithm is as follows. Consider a market where none of the sellers is aware of the equilibrium policies for the current round of the game. Each seller solves the best response problem by observing her competitors' prices from the previous realization of the game and adopts a policy that maximizes her payoff. No seller has information about the inventories of her competitors except the prices realized in the previous round of the game. The entire multi-period game is repeated for a number of times until the market approaches an equilibrium state. We denote the best response problem seller i solves with respect to her competitors' strategies from the kth round of game by $BR_i((\mathbf{p}_{-i})^k)$. The equilibrium prices are denoted by $\mathbf{p}^*$. The iterative learning algorithm is formally presented as follows:

Iterative Learning Algorithm:

for $i \in \mathbf{I}$ **do**
    $(p_i^t)^0 \leftarrow p_{initial}^t$
end for
for $i \in \mathbf{I}$ **do**
    $(\mathbf{p}_i)^1 \leftarrow BR_i((\mathbf{p}_{-i})^0)$
end for
$k \leftarrow 1$
while $(\mathbf{p}_i)^k \neq (\mathbf{p}_i)^{k-1}$ **do**
    for $i \in \mathbf{I}$ **do**
        $(\mathbf{p}_i)^{k+1} \leftarrow BR_i((\mathbf{p}_{-i})^k)$
    end for
    $k \leftarrow k + 1$
end while
$\mathbf{p}^* \leftarrow (\mathbf{p})^k$

15: **RETURN** p^*

5.3 Implementation issues

We implement the iterative learning algorithm for both a two-period and a three-period pricing problem. We use the dynamic programming model to solve the best response problem at each iteration. However, as the dynamic programming model in our pricing problem generates a look-up table of optimal prices for each discretized inventory value, and the inventory is affected by the actual realization of the uncertain parameters, we need to generate realized values for the uncertain parameters in the simulation in order for each seller's competitors to observe her prices at the previous iteration. This introduces a problem: if we randomly generate realized values for the uncertain parameters, the pricing policy each seller adopts based on her look-up table generated by the dynamic programming model will be random too. Therefore, each seller's pricing policy is not only affected by her competitors's prices from the previous iteration, but also by the realized values of the uncertain parameters at previous iteration we have simulated. If the realized values of the uncertain parameters are generated randomly, sellers' prices can hardly converge to a steady state. This implies that the market equilibrium can never be reached using this algorithm.

In order to address this problem, we try to generate a large number of realized values for the uncertain parameters, find the deterministic pricing policy for each of them, and take the average of all the pricing policies as the policy that will be observed by her competitors at the following iteration. When the number of realized values we generated for the uncertain parameters at each iteration is very large, the resulting pricing policy becomes less affected by the uncertain parameters, and therefore, the market equilibrium may be computed. Our computational results (from Figure B-14 to FB-17) show that 10 realized values generated for the uncertain parameters at each iteration are good enough to ensure the convergence of the iterative learning algorithm.

5.4 Computational results

First, we consider a duopoly market and study if the iterative learning algorithm converges to an equilibrium pricing policy. We use test1.dat as the test instance in this case. We tried different number of realized values generated for the uncertain parameters at each iteration. Therefore, we can compare how the number of realized values generated will affect the convergence of the prices. We also try four different initial prices to see how sensitive the iterative learning algorithm is to the initial prices chosen. We can see from Figure B-14 to Figure B-17 that the prices converge when the number of realized values generated for the uncertain parameters is large. Furthermore, for all the four initial prices, the equilibrium prices are around (1.7, 6.2) for seller 1 and (2.7, 3.3) for seller 2. Note that even when the initial price we choose is (7,8) for seller 1 and (4,7) for seller 2, which are both higher than the converged price of these two sellers respectively, the iterative learning algorithm is able to compute the equilibrium prices. This shows that the iterative learning algorithm is not very sensitive to the choice of initial prices.

Next, we consider a market with three sellers. Similarly, we try a different number of realized values generated for the uncertain parameters at each iteration. We try four different initial prices to see how sensitive the iterative learning algorithm is to the initial prices. We can see from Figure B-18 to Figure B-21 that the prices converge when the number of realized values generated for the uncertain parameters is 10. Furthermore, for all the four initial prices, the equilibrium prices are around (1.6, 6.2) for seller 1, (2.4, 3.3) for seller 2 and (3.1, 3.1) for seller 3.

5.5 Convergence of the iterative learning algorithm

The reason that the iterative learning algorithm converges to a Nash equilibrium for Robust-M1 is that Robust-M1 fixes the demand to be the worst-case demand, but not the demand decided by nature (i.e., the demand function with the actual realization of uncertain parameters). As we can identify that the worst-case scenario

can only happen at an extreme point, we, in fact, transform the robust demand model to a deterministic demand model. In other words, we get rid of the side effect of uncertainty on the profit in this model. Therefore, a Nash equilibrium exists and is computable, as the uncertainty factor no longer affects the solution.

For the rest of the models, such as the dynamic programming formulation, the iterative learning algorithm can still reach a Nash equilibrium policy if we take enough actual realizations of uncertain parameters, and use the average of them to decide the prices. This is, in fact, another way to minimize the effect of uncertainty on the solution. Numerical results show that a Nash equilibrium can be computed if the number of actual realizations of uncertain parameters is large enough (such as 10 sets of realizations).

Chapter 6

Conclusions

In this thesis, we formulated a multi-period pricing model for a monopoly and an oligopoly market. Each seller has a fixed starting inventory, and the products are perishable. The demand function faced by each seller is uncertain and is dependent on all the sellers' prices at the current period and the past pricing history. Our focus was to address two of the main issues in the pricing model: uncertainty and competition.

To address the issue of uncertainty in demand, we first studied the pricing problem in a monopoly setting. We used robust optimization, adaptable robust optimization, dynamic programming and stochastic optimization to formulate various dynamic pricing models, and find an adaptable closed pricing policy for each of these models. We examined the effect of budget of uncertainty, reference price, delayed resource allocation, and feedback control on the quality of the pricing policy. We later extended the robust models to a multi-period setting and show that the results found in a monopoly and two-period setting are still valid.

We addressed the issue of competition by studying if a market equilibrium exists in our models. We also used an iterative learning algorithm to compute the equilibrium prices. We showed through numerical results that the algorithm computes the market equilibrium prices if we generate a significantly large number of realized values for the uncertain parameters at each step of the iterative learning algorithm.

There are a number of interesting findings from this research, which are discussed

as follows:

1. Through the comparison of Robust-M1 and Robust-M2, we found that the consideration of the reference price in the demand function has a positive effect on the quality of the resulting pricing policy. Thus, the presence of the reference price not only makes the demand function a better model of customers' attitudes towards purchasing, but also increases the overall profit that can be obtained.

2. For the same robust optimization model, an increase in the budget of uncertainty will cause a decrease in the objective value. This is because a higher budget of uncertainty corresponds to a bigger uncertainty set, which will make the worst-case solution even worse. However, an increase in the budget of uncertainty also causes a decrease in the probability of constraint violation, as the protection level increases.

3. From the comparison of Robust-M1 with Robust-M3 and Robust-M3-AARC, we found that there are both advantages and disadvantages of the delayed resource allocation adopted in Robust-M3 and Robust-M3-AARC. The advantage of the delayed resource allocation is to give models the flexibility to change their policies with respect to the actual realization of uncertain parameters. As a result, the objective values of Robust-M3 and Robust-M3-AARC are higher than that of Robust-M1 on average. However, the disadvantage is that it leads to a lower objective value in the worst-case scenario, as the choices of prices are more restricted in these two models in order to protect the resource capacity constraint against all possible realization of uncertain parameters.

4. By comparing Robust-M3 with Robust-M3-AARC, we found that a closed-loop policy outperforms an open-loop policy in general. The feedback control in Robust-M3-AARC acts as a tuning effect that tries to tune the current decision based on the uncertain parameters that have realized. However, this tuning effect is just an estimation of what could happen in reality, and it does not guarantee an increase in the objective value at all times.

5. We compared the dynamic programming model with robust optimization models and found that the dynamic programming model performs better than robust optimization models in general. However, when a feedback control is used in the robust formulation, the results obtained by the dynamic programming model and Robust-M3-AARC are very close. In fact, we have shown in Chapter 3 that Robust-M3-AARC even outperforms the dynamic programming model for as high as 28% of the total sample points in our simulation. This is a very promising result. It shows that a robust formulation with feedback control can achieve almost as good results as a dynamic programming formulation. Moreover, Robust-M3-AARC is computationally tractable and, yet, produces good results in a multi-period setting.

6. The stochastic optimization model gives a higher profit than that of robust optimization models. Similar to the robust optimization model, the stochastic optimization model also benefits from the pre-computation of the optimal solution for each scenario.

7. In a multi-period setting, dynamic programming and stochastic optimization lose to robust optimization, as they either suffer from the curse of dimensionality in the state variable or the increase in the number of periods. In contrast, robust optimization is able to solve the large-scale optimization problem. We used a delayed constraint generation method to solve the robust optimization models in the multi-period setting.

8. We studied a Nash equilibrium in an oligopoly setting. We showed the existence of solution for both the best response subproblems and the market equilibrium problem for all of our models. However, for all of our models except Robust-M1, as the profit obtained by each of them is affected by the actual realization of uncertain parameters, an equilibrium pricing policy may not be as easily computable. We considered an iterative learning algorithm and made some assumptions to minimize the effect of uncertainty on these models. We showed using simulation how this algorithm computes a Nash equilibrium policy.

Appendix A

Tables

Table A.1: Effect of reference price: comparison of the profits obtained by Robust-M1 and Robust-M2.

	test1	test2	test3	test4	test5
Robust-M1	109.37	7	107.48	137.603	109.37
Robust-M2	112.79	8.4426	116.18	152.83	112.79

Table A.2: The profit obtained and the ratio of constraint violation for different budget of uncertainty

	test4	test6	test7	test8	test9	test10	test11	test12	test13
Profit	137.6	79.9	49.9	18	12.4	7.9	4.4	1.7	0.4
Ratio	45.0%	41.7%	36.8%	24.6%	23.1%	12.3%	11.1%	4.7%	3.9%

Table A.3: The profits obtained by Robust-M1, Robust-M3 and Robust-M3-ARRC for different test instances.

	test1	test2	test3	test4	test5
Robust-M1	109.37	7	107.48	137.603	109.375
Robust-M3	109.37	2	107.48	137.603	75
Robust-M3-ARRC	109.37	2	107.48	137.603	75

Table A.4: The percentage of sample points for which Robust-M1 has a higher profit than Robust-M3 and Robust-M3-AARC respectively.

	test1	test2	test3	test4	test5
Robust-M3	0%	0.55%	0%	0%	13.21%
Robust-M3-AARC	0%	1.46%	0%	0%	13.21%

Table A.5: The percentage of sampling points for which Robust-M3 has a higher profit than that of Robust-M3-AARC.

	test1	test2	test3	test4	test5
% of sampling points	21.1%	8.62%	14.49%	24.77%	0%

Table A.6: The percentage of sample points for which Robust-M3-AARC has a higher profit than that of the dynamic programming model.

	test1	test2	test3	test4	test5
% of sampling points	19%	2%	13%	12%	28%

Table A.7: The profits obtained by Robust-M1, Robust-M3, Robust-M3-ARRC and stochastic optimization model for different test instances.

	test1	test2	test3	test4	test5
Robust-M1	109.37	7	107.48	137.603	109.375
Robust-M3	109.37	2	107.48	137.603	75
Robust-M3-ARRC	109.37	2	107.48	137.603	75
Stochastic programming	182.759	197.64	131.226	177.35	182.75

Table A.8: The percentage of sample points that lead to lower objective values than that obtained by Robust-M3-AARC.

	test1	test2	test3	test4	test5
Percentage	0%	2.5%	0%	0%	0.015%

Table A.9: The running time required by the dynamic programming model to solve the pricing problem for various numbers of time periods.

Period	CPU Time
2	1.26s
3	235.43s
4	$\geq$30min

Table A.10: Effect of the reference price: comparison of profits obtained by Robust-M1 and Robust-M2 on the eight-period test instances.

	test8-1	test8-2	test8-3
Robust-M1	654.29	439.23	275.36
Robust-M2	692.38	482.15	314.83

Appendix B

Figures

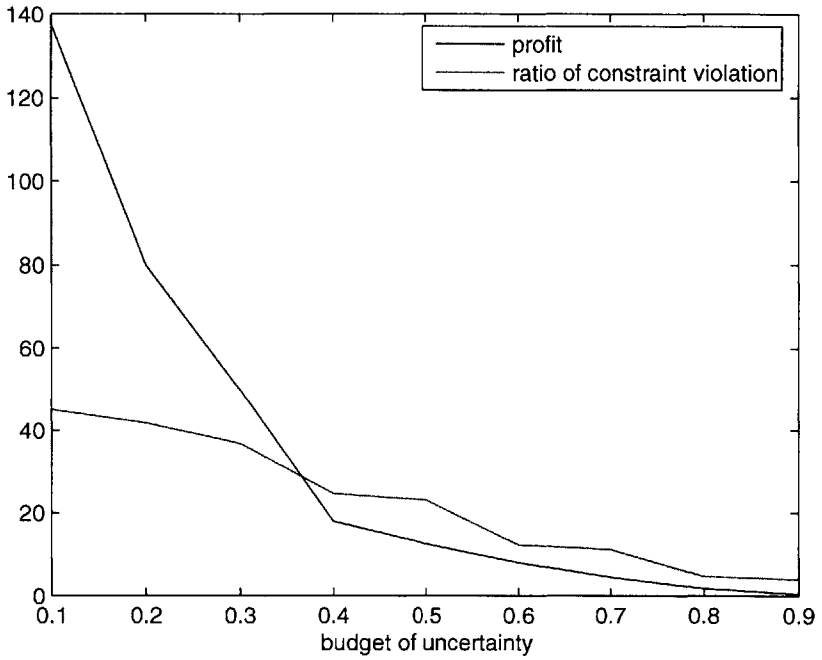


Figure B-1: Effect of budget of uncertainty: The profits obtained and the probability of constraint violation for different budget of uncertainty in a monopoly and two-period setting.

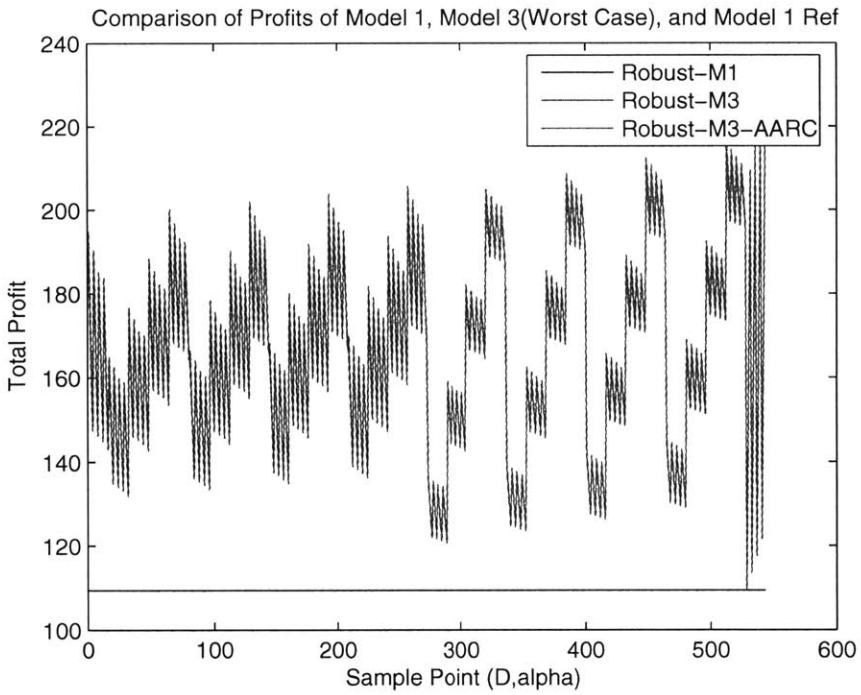


Figure B-2: Effect of delayed resource allocation: The profits obtained by Robust-M1, Robust-M3 and Robust-M3-AARC for various realizations of uncertain parameters on test1.dat in a monopoly and two-period setting.

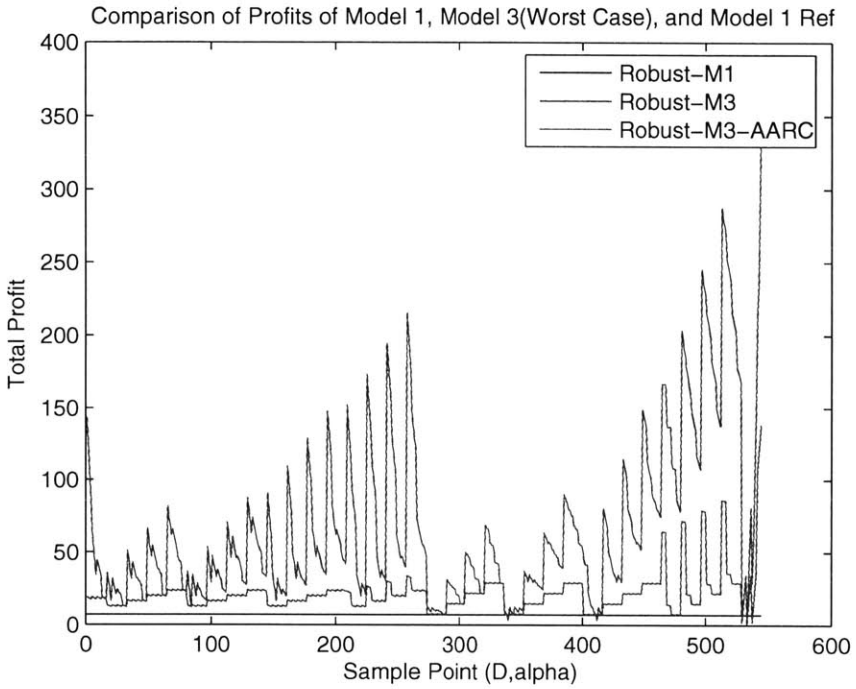


Figure B-3: Effect of delayed resource allocation: The profits obtained by Robust-M1, Robust-M3 and Robust-M3-AARC for various realizations of uncertain parameters on test2.dat in a monopoly and two-period setting.

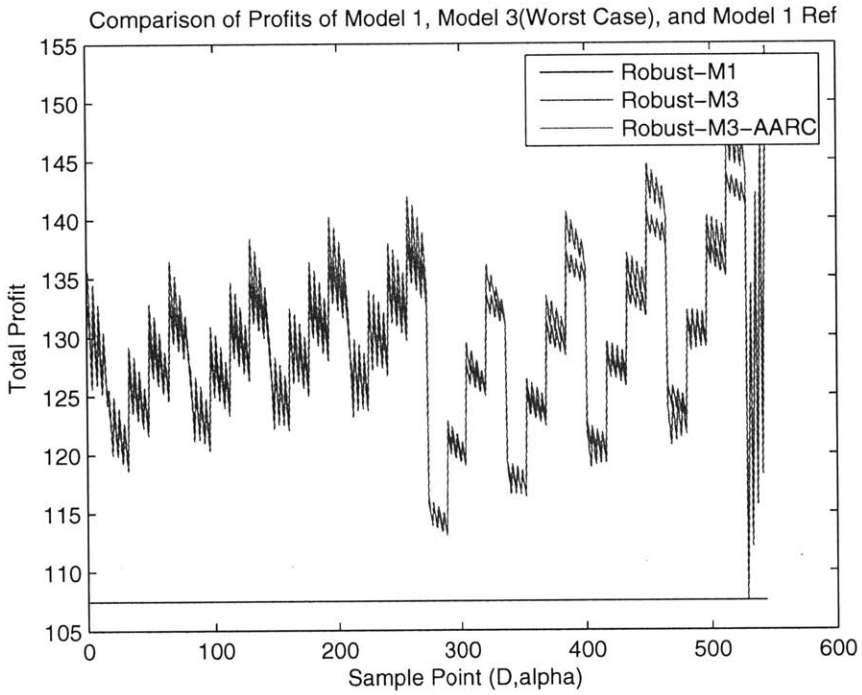


Figure B-4: Effect of delayed resource allocation: The profits obtained by Robust-M1, Robust-M3 and Robust-M3-AARC for various realizations of uncertain parameters on test3.dat in a monopoly and two-period setting.

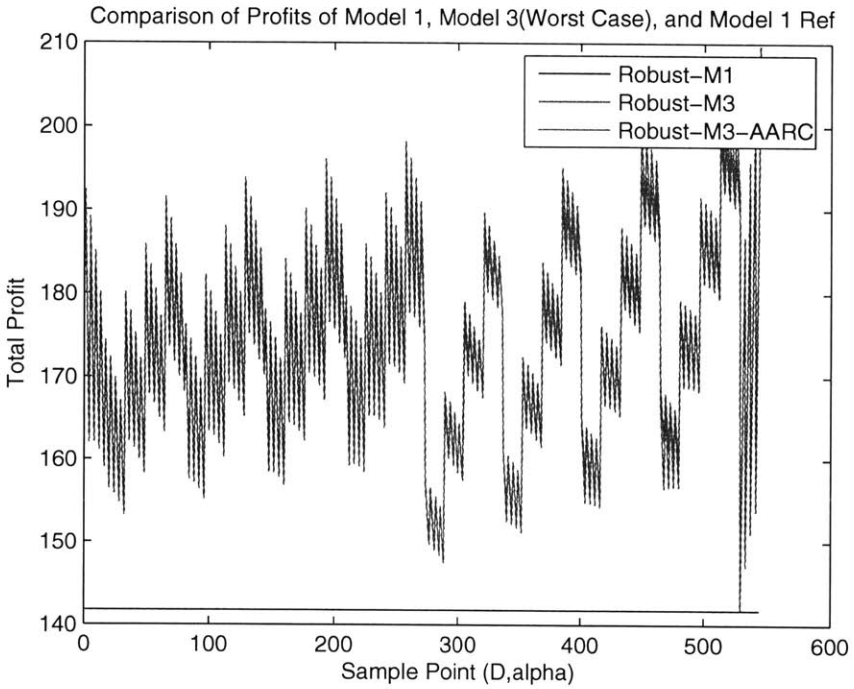


Figure B-5: Effect of delayed resource allocation: The profits obtained by Robust-M1, Robust-M3 and Robust-M3-AARC for various realizations of uncertain parameters on test4.dat in a monopoly and two-period setting.

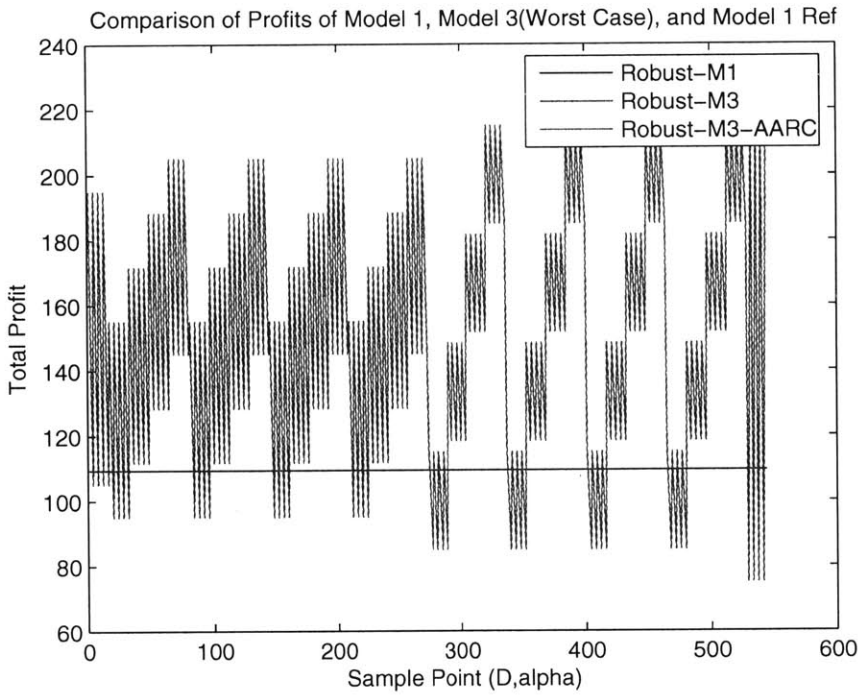


Figure B-6: Effect of delayed resource allocation: The profits obtained by Robust-M1, Robust-M3 and Robust-M3-AARC for various realizations of uncertain parameters on test5.dat in a monopoly and two-period setting.

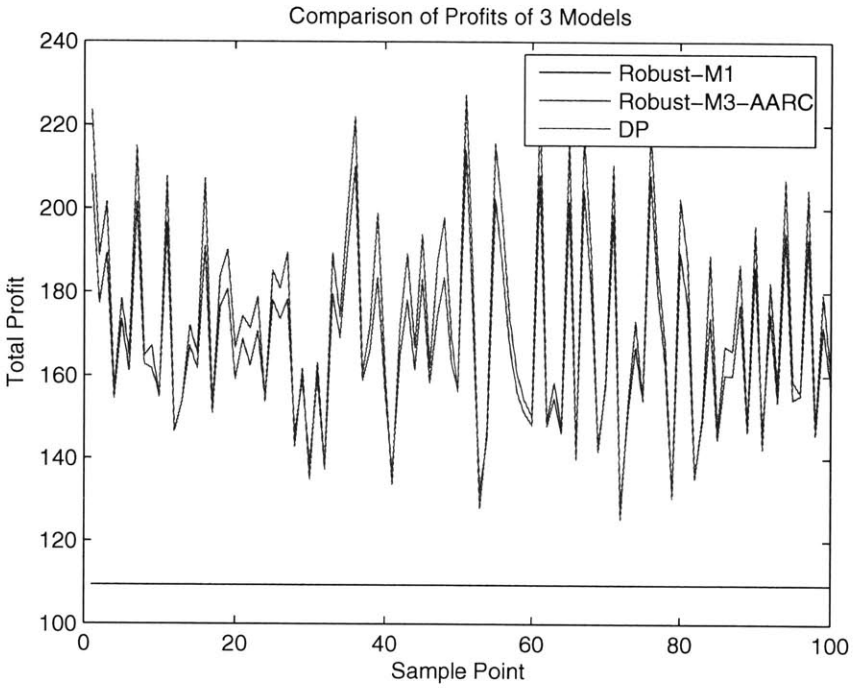


Figure B-7: Comparison of the profits obtained by Robust-M1, Robust-M3-AARC and dynamic programming model for various realizations of uncertain parameters on test1.dat in a monopoly and two-period setting.

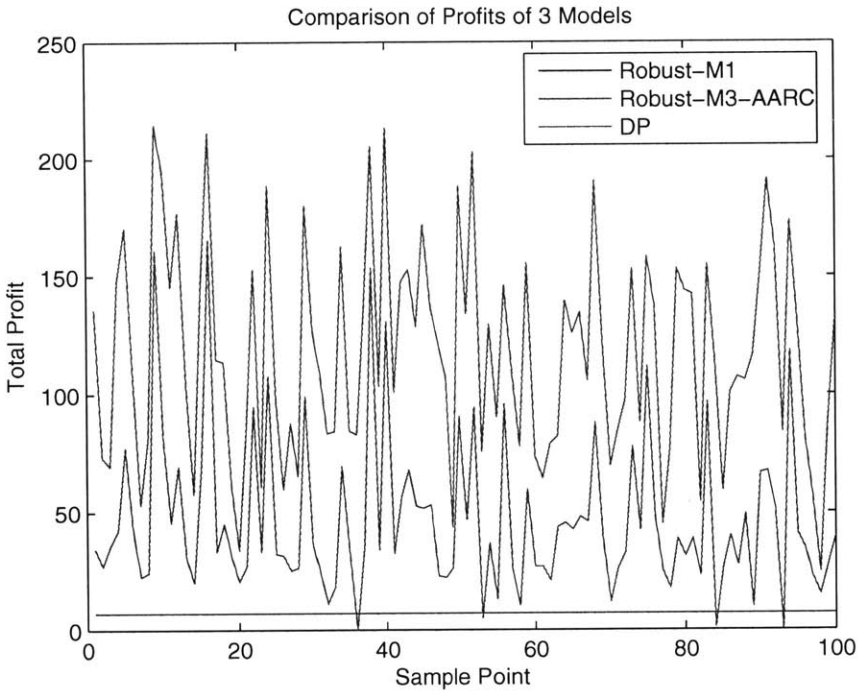


Figure B-8: Comparison of the profits obtained by Robust-M1, Robust-M3-AARC and dynamic programming model for various realizations of uncertain parameters on test2.dat in a monopoly and two-period setting.

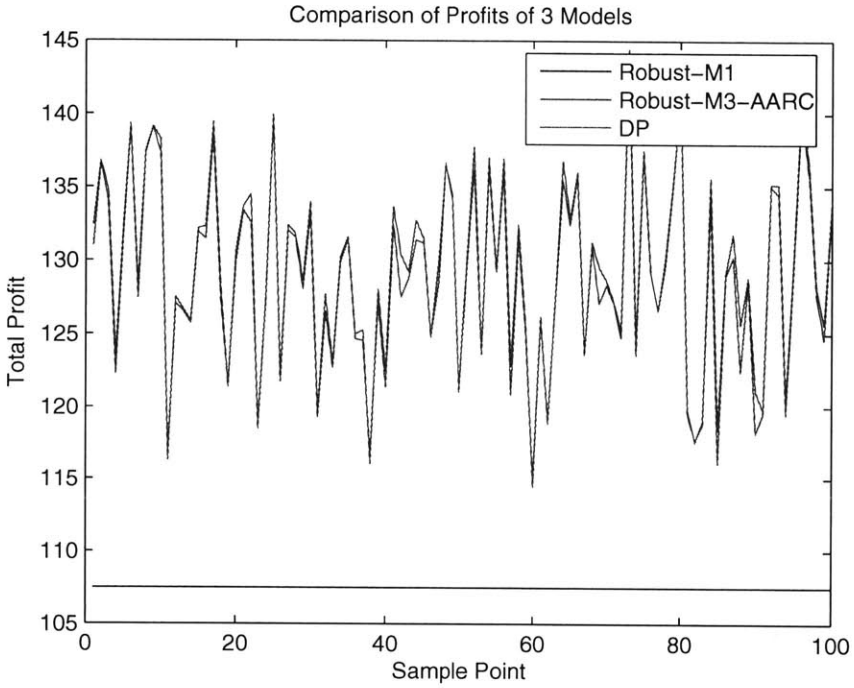


Figure B-9: Comparison of the profits obtained by Robust-M1, Robust-M3-AARC and dynamic programming model for various realizations of uncertain parameters on test3.dat in a monopoly and two-period setting.

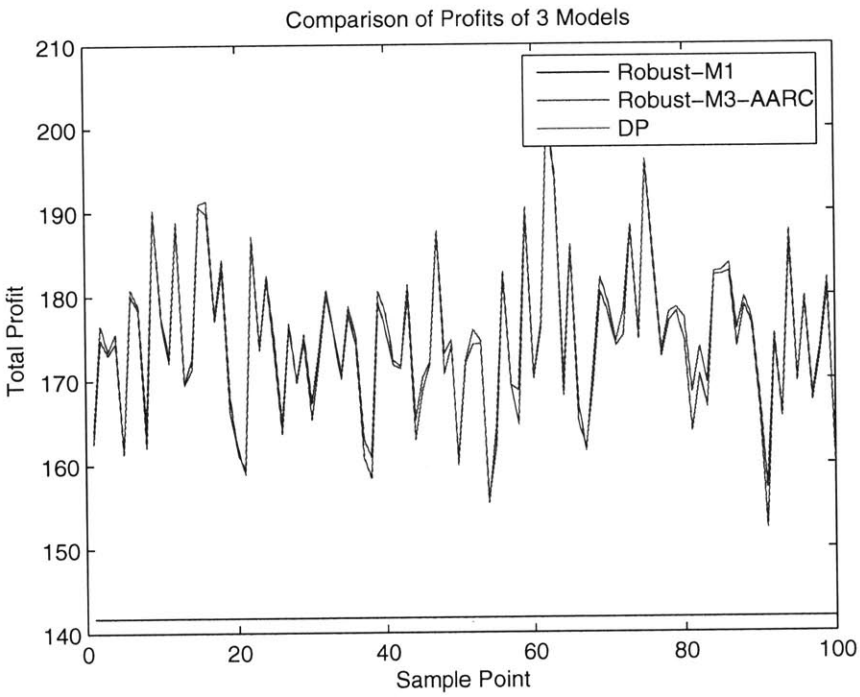


Figure B-10: Comparison of the profits obtained by Robust-M1, Robust-M3-AARC and dynamic programming model for various realizations of uncertain parameters on test4.dat in a monopoly and two-period setting.

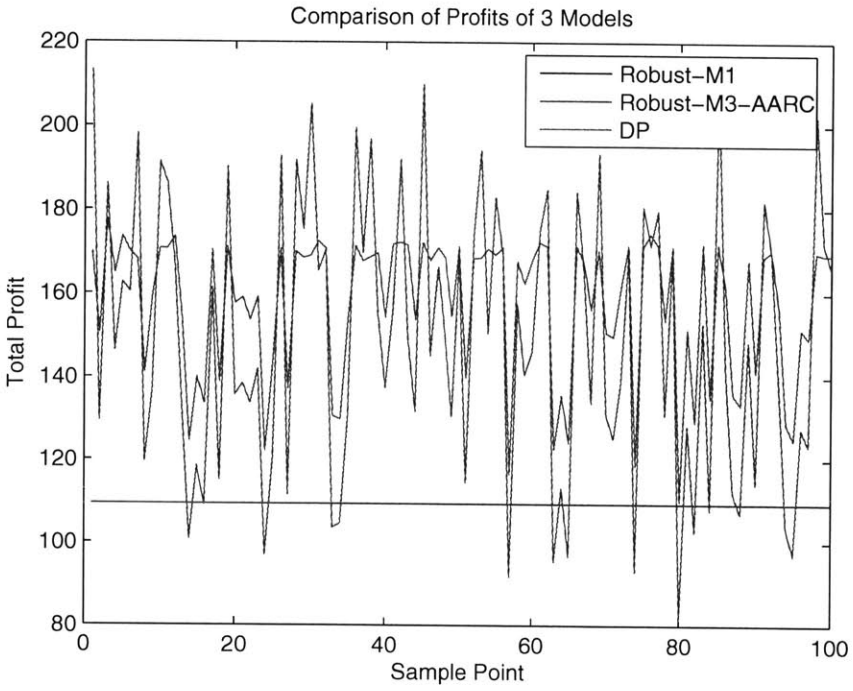


Figure B-11: Comparison of the profits obtained by Robust-M1, Robust-M3-AARC and dynamic programming model for various realizations of uncertain parameters on test5.dat in a monopoly and two-period setting.

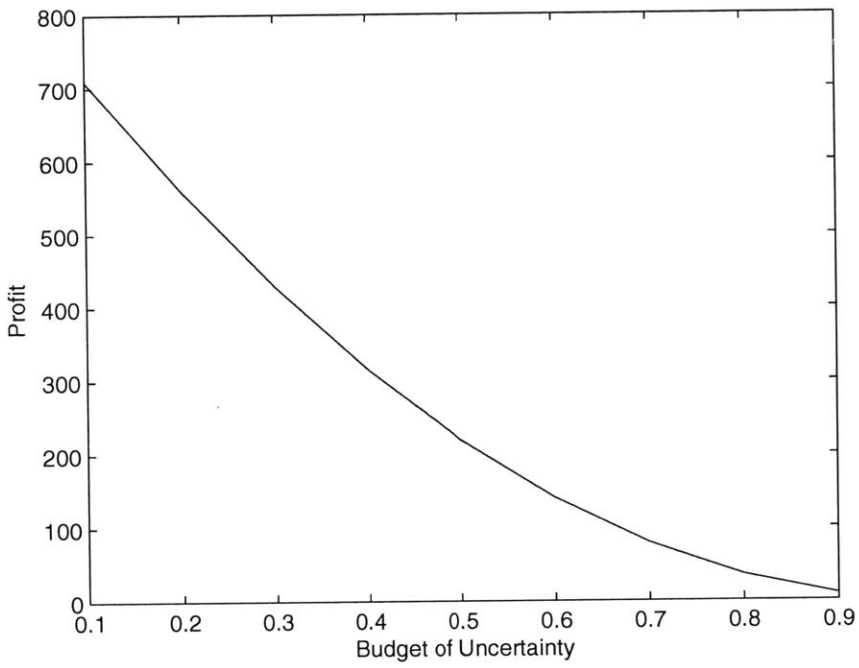


Figure B-12: Comparison of the profits obtained by Robust-M1 by varying the budget of uncertainty on test8-1.dat in a monopoly and eight-period setting.

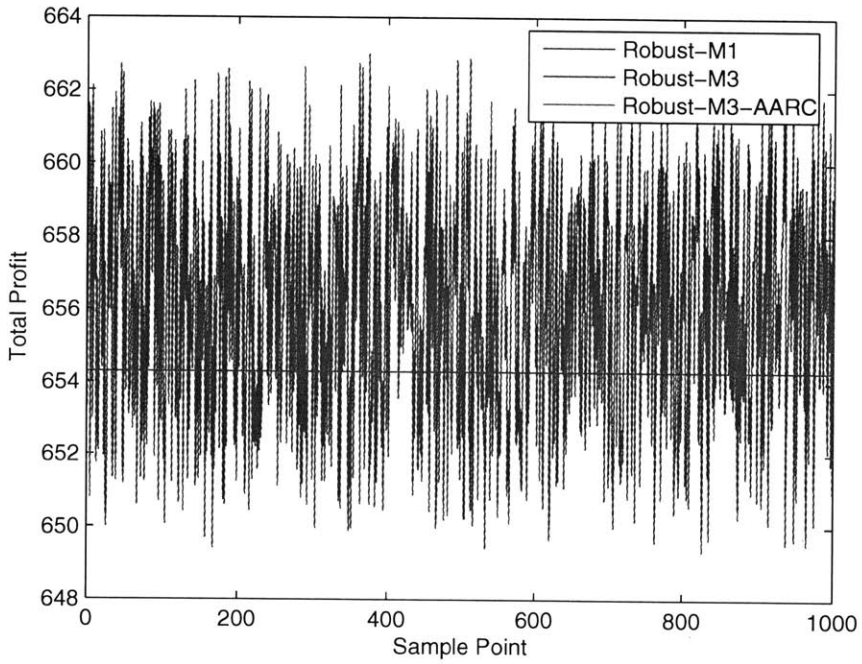
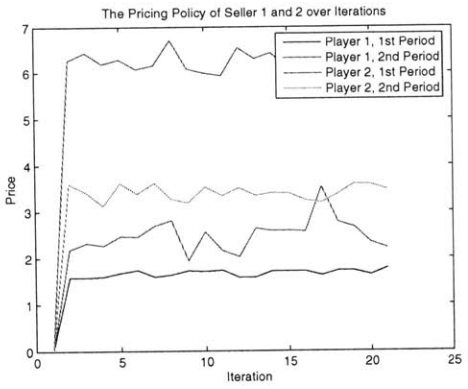
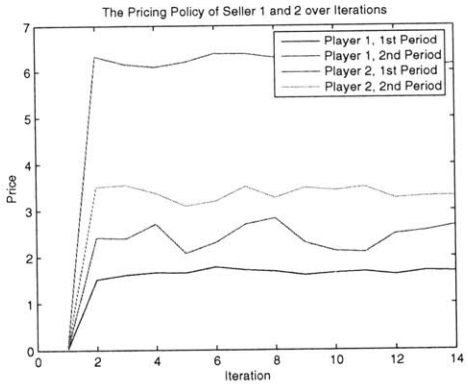


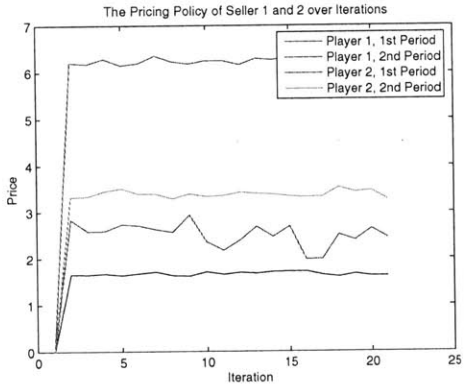
Figure B-13: Comparison of the profits obtained by Robust-M1, Robust-M3 and Robust-M3-AARC on test8-1.dat in a monopoly and eight-period setting.



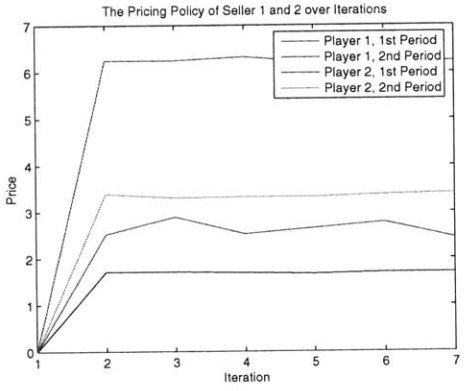
(a) 2 sets of realized values generated



(b) 3 sets of realized values generated

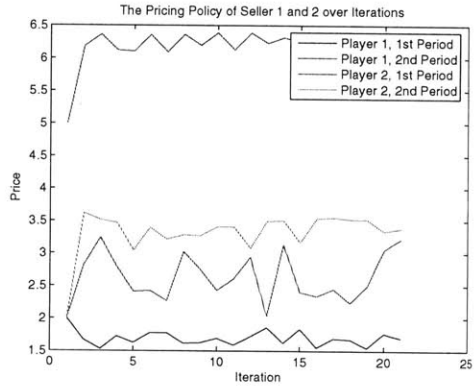


(c) 5 sets of realized values generated

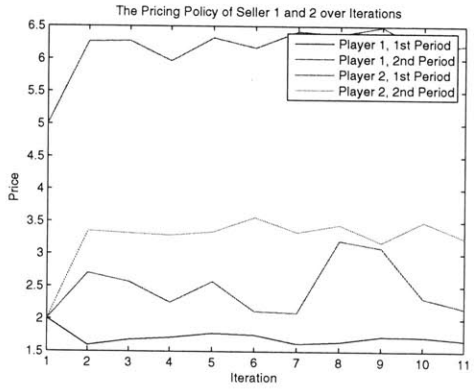


(d) 10 sets of realized values generated

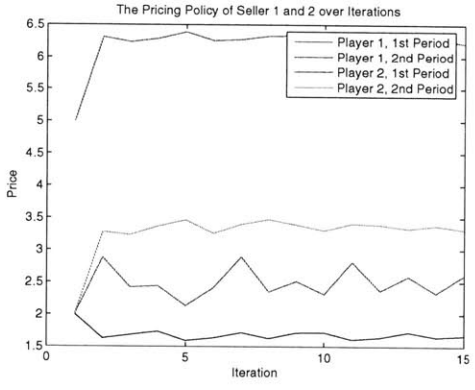
Figure B-14: The pricing policies of two sellers at different iterations of the iterative learning process. The initial price is (0,0) for both sellers, and a different numbers of sets of realized values are generated for the uncertain parameters.



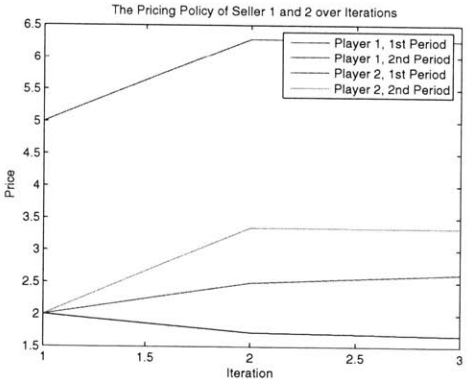
(a) 2 sets of realized values generated



(b) 3 sets of realized values generated

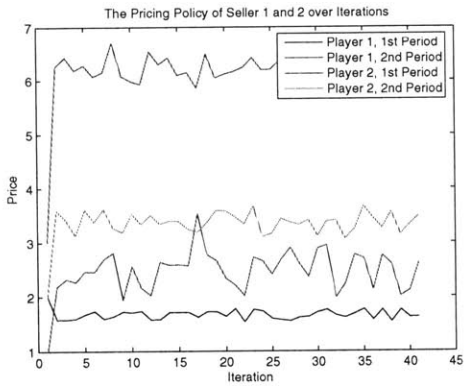


(c) 5 sets of realized values generated

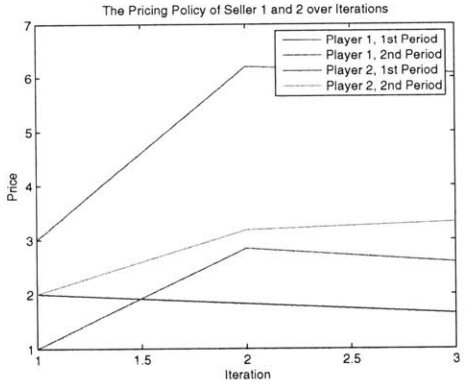


(d) 10 sets of realized values generated

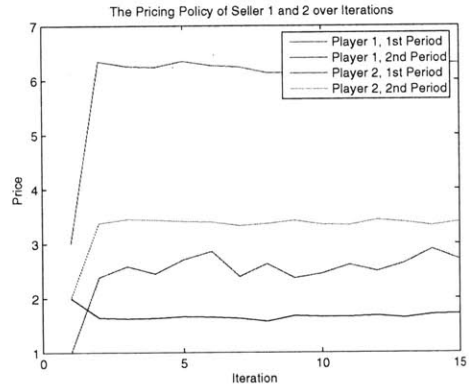
Figure B-15: The pricing policies of two sellers at different iterations of the iterative learning process. The initial price is (2,5) for seller 1 and (2,2) for seller 2, and a different numbers of sets of realized values are generated for the uncertain parameters.



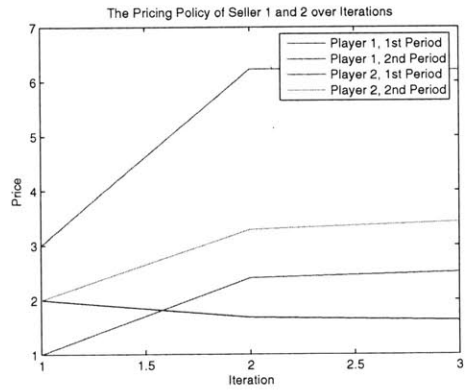
(a) 2 sets of realized values generated



(b) 3 sets of realized values generated

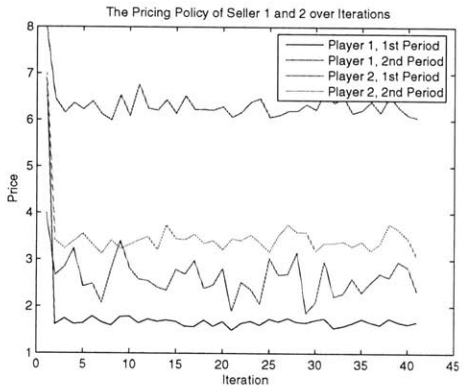


(c) 5 sets of realized values generated

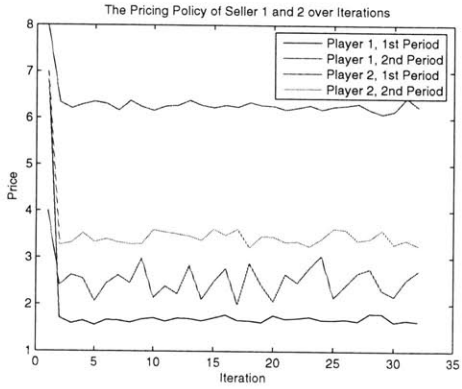


(d) 10 sets of realized values generated

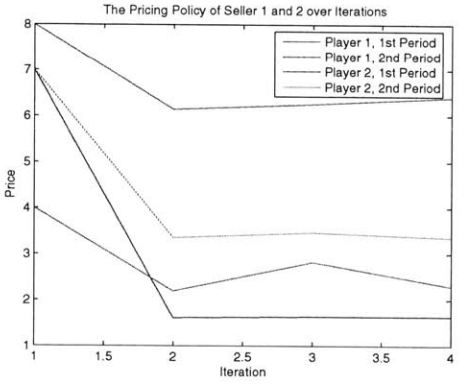
Figure B-16: The pricing policies of two sellers at different iterations of the iterative learning process. The initial price is (2,3) for seller 1 and (1,2) for seller 2, and a different numbers of sets of realized values are generated for the uncertain parameters.



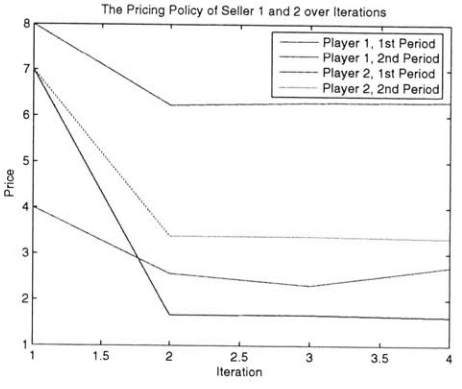
(a) 2 sets of realized values generated



(b) 3 sets of realized values generated

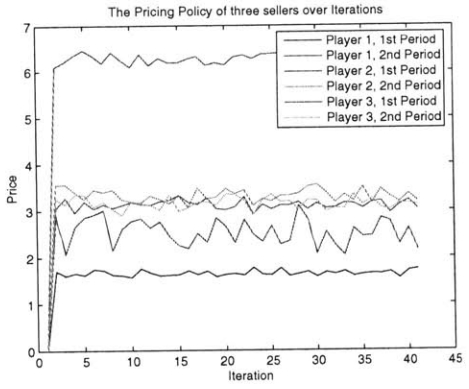


(c) 5 sets of realized values generated

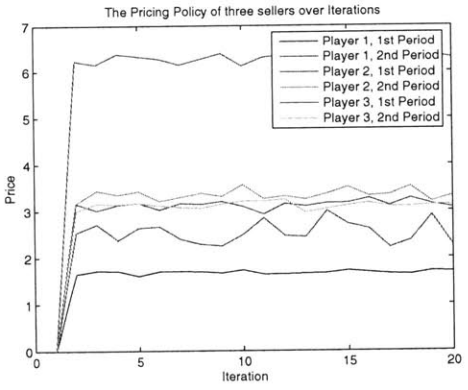


(d) 10 sets of realized values generated

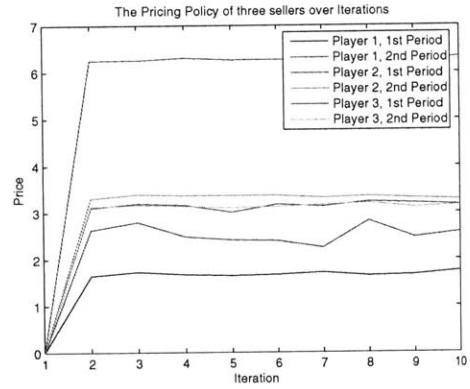
Figure B-17: The pricing policies of two sellers at different iterations of the iterative learning process. The initial price is (7,8) for seller 1 and (4,7) for seller 2, and a different number of sets of realized values are generated for the uncertain parameters.



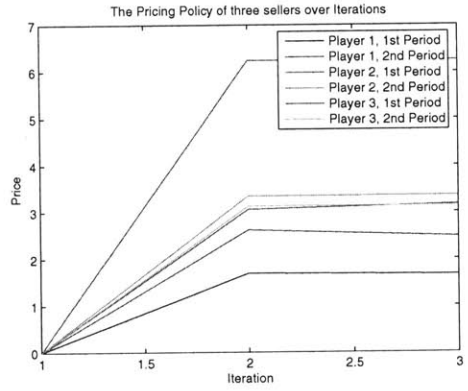
(a) 2 sets of realized values generated



(b) 3 sets of realized values generated

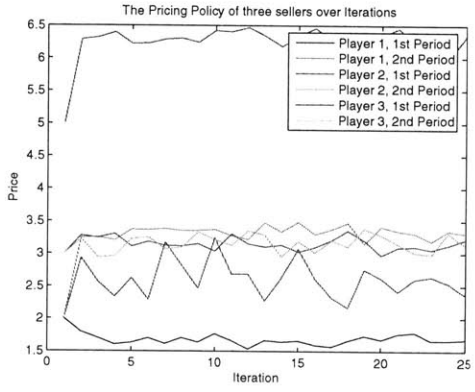


(c) 5 sets of realized values generated

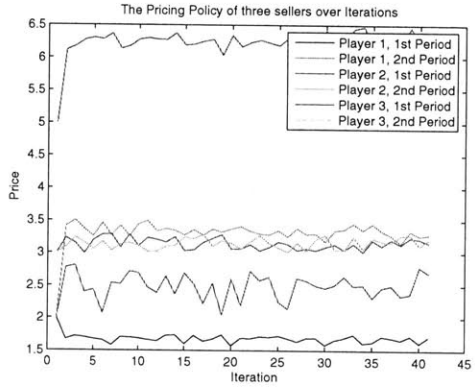


(d) 10 sets of realized values generated

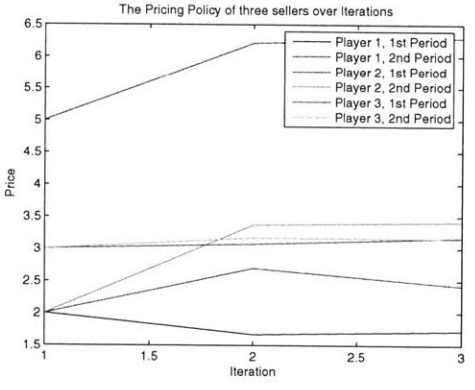
Figure B-18: The pricing policies of three sellers at different iterations of the iterative learning process. The initial price is (0,0) for all the sellers, and a different number of sets of realized values are generated for the uncertain parameters.



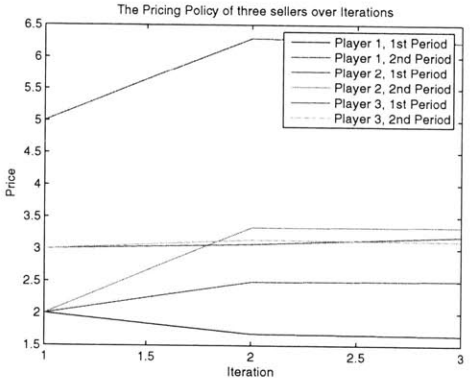
(a) 2 sets of realized values generated



(b) 3 sets of realized values generated

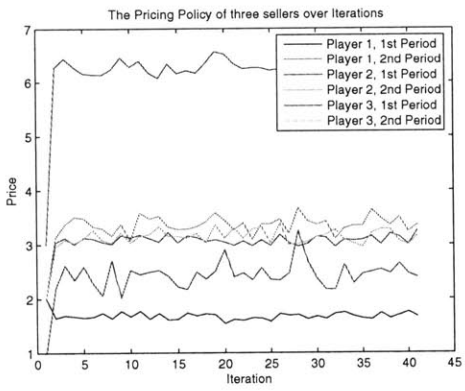


(c) 5 sets of realized values generated

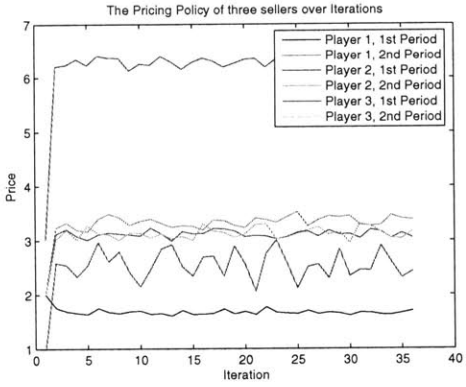


(d) 10 sets of realized values generated

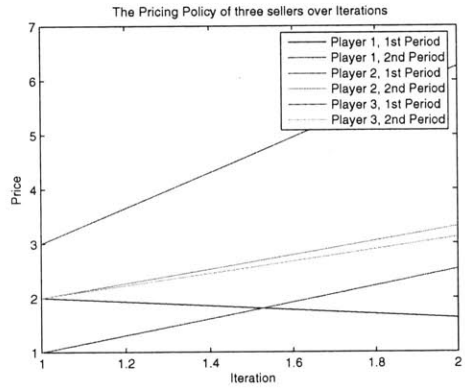
Figure B-19: The pricing policies of three sellers at different iterations of the iterative learning process. The initial price is (2,5) for seller 1, (2,2) for seller 2 and (3,3) for seller 3, and a different number of sets of realized values are generated for the uncertain parameters.



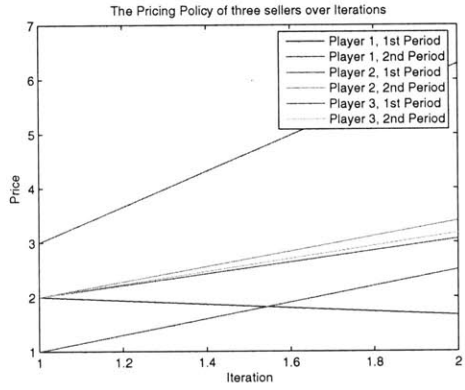
(a) 2 sets of realized values generated



(b) 3 sets of realized values generated

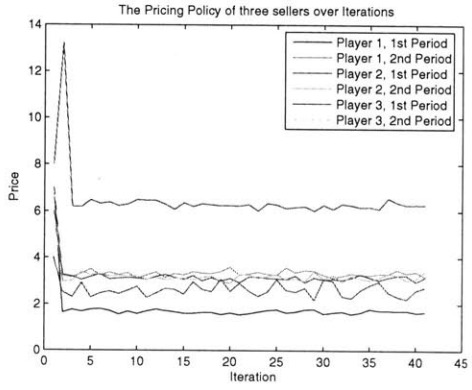


(c) 5 sets of realized values generated

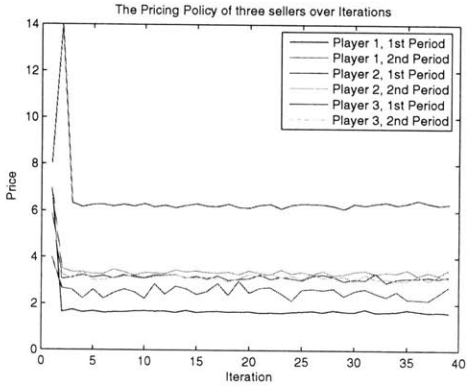


(d) 10 sets of realized values generated

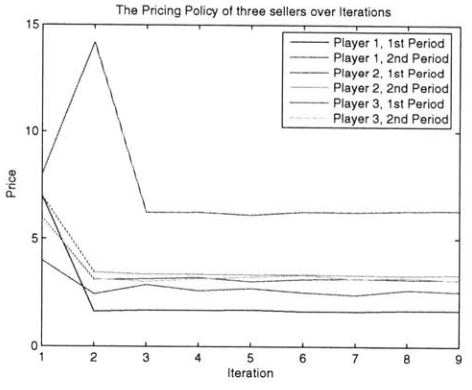
Figure B-20: The pricing policies of three sellers at different iterations of the iterative learning process. The initial price is (2,3) for seller 1, (1,2) for seller 2 and (2,2) for seller 3, and a different number of sets of realized values are generated for the uncertain parameters.



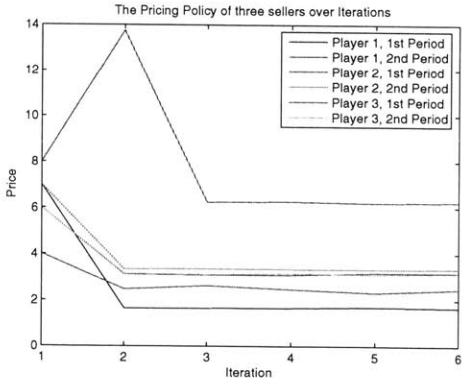
(a) 2 sets of realized values generated



(b) 3 sets of realized values generated



(c) 5 sets of realized values generated



(d) 10 sets of realized values generated

Figure B-21: The pricing policies of three sellers at different iterations of the iterative learning process. The initial price is (7,8) for seller 1, (4,7) for seller 2 and (6,6) for seller 3, and a different number of sets of realized values are generated for the uncertain parameters.

Bibliography

[1] E. Adida. *Dynamic Pricing and Inventory Control with no Backorders under Uncertainty and Competition.* PhD thesis, Massachusetts Institute of Technology, 2006.

[2] W.L. Baker, E. Lin, M.V. Marn, and C.C. Zawada. Getting prices right on the web. *The McKinsey Quarterly*, (2), 2001.

[3] M.S. Bazaraa, H.D. Sherali, and C. Shetty. *Nonlinear Programming.* John Wiley & Sons, New York, 1993.

[4] A. Ben-Tal, A. Goryashko, E. Guslitzer, and A. Nemirovski. Adjustable robust solutions of uncertain linear programs. *Mathematical Programming*, (99):351–376, 2004.

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

[6] A. Ben-Tal and A. Nemirovski. Robust solutions to uncertain linear programs. *Operations Research letters*, (25):1–13, 1999.

[7] D. Bertsimas and M. Sim. The price of robustness. *Operations Research*, (52):25–53, 2001.

[8] E. Cavazzuti, M. Pappalardo, and M. Passacantando. Nash equilibria, variational inequalities, and dynamic systems. *Journal of Optimization Theory and Applications*, (114):491–506, 2002.

[9] X. Chen, M. Sim, P. Sun, and J.W. Zhang. A tractable approximation of stochastic programming via robust optimization. Working paper, 2006.

[10] P. Cubiotti. Generalized quasi-variational inequalities in infinite-dimensional normed spaces. *Journal of Optimization Theory and Applications*, (92):457–475, 1997.

[11] P. Cubiotti and N.D. Yen. A result related to ricceri's conjecture on generalized quasi-varitional inequalities. *Archiv der Mathematik*, (69):507–514, 1997.

[12] D. Fudenberg and J. Tirole. *Dynamic Models of Oligopoly.* Harwood Academic Publishers, New York, 1986.

[13] D. Fudenberg and J. Tirole. *Game Theory.* The MIT Press, Cambridge, 1991.

[14] C. Gaimon. Dynamic game results of the acquisition of new technology. *Operations Research*, (37), 1989.

[15] D. Goldfarb and G. Iyengar. Robust porfolio selection problems. *Mathematics of Operation Research*, (28), 1994.

[16] A.P. Kirman and M.J. Sobel. Dynamic oligopoly with inventories. *Econometrica: Journal of the Econometric Society*, (42):279–288, 1974.

[17] G. Perakis and E. Adida. A robust optimization approach to dynamic pricing and inventory control with no backorders. *Mathematical Programming Special Issue on Robust Optimization*, 2006.

[18] G. Perakis and D.T. Nguyen. Robust competitive pricing for multiple perishable products. Working paper, 2005.

[19] I. Popescu and Y. Wu. Dynamic pricing strategies with reference effects. *Operations Research*, 2005.

[20] J.B. Rosen. Existence and uniqueness of equilibrium points for concave n-person games. *Econometrica: Journal of the Econometric Society*, (33):520–534, 1965.

[21] A. Sood. *Competitive Multi-period Pricing for Perishable Products.* PhD thesis, Massachusetts Institute of Technology, 2004.

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

[23] K. Talluri and G. van Ryzin. *The Theory and Practice of Revenue Management.* Kluwer Academic Publishers, 2004.

[24] S. Uryas'ev and R.Y. Rubinstein. On relaxation algorithms in computation of noncooperative equilibria. *IEEE Transaction on Automatic Control*, (39):1263–1267, 1994.

[25] X. Vives. *Oligopoly Pricing. Old Ideas and New Tools.* MIT Press, Cambridge, 1999.

[26] E. Zabel. Monopoly and uncertainty. *The Review of Economic Studies*, (37):205–219, 1970.