DYNAMIC LEARNING MECHANISMS IN REVENUE MANAGEMENT PROBLEMS

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF MANAGEMENT
SCIENCE AND ENGINEERING
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Zizhuo Wang
May 2012

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Yinyu Ye, Primary Adviser

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Peter Glynn

I certify that I have read this dissertation and that, in my opinion, it is fully adequate in scope and quality as a dissertation for the degree of Doctor of Philosophy.

Tze Lai

Approved for the Stanford University Committee on Graduate Studies.

Patricia J. Gumport, Vice Provost Graduate Education

This signature page was generated electronically upon submission of this dissertation in electronic format. An original signed hard copy of the signature page is on file in University Archives.

Abstract

One of the major challenges in revenue management problems is the uncertainty of the customer's demand. In many practical problems, the decision maker can only observe realized demand over time, but not the underlying demand. Methods to incorporate demand learning into revenue management problems remain an important topic both for academic researchers and practitioners.

In this dissertation, we propose a class of dynamic learning mechanisms for a broad range of revenue management problems. Unlike methods that attempt to deduce customer demand before the selling season, our proposed mechanisms enable a decision maker to learn the demand "on the fly". That is, they integrate learning and doing into a concurrent procedure and both learning and doing are accomplished during the selling season. Furthermore, we show that our mechanisms have the following features:

1. *Robustness*, our mechanisms work well for a wide range of underlying demand structures and do not need to know the structure beforehand;

2. *Dynamism*, our mechanisms take advantages of increasing sales data as selling season moves forward and update the demand information along the process;

3. *Optimality*, our mechanisms achieve near-optimal revenues, as compared to a clairvoyant who has complete knowledge of the demand information beforehand.

By comparing with different learning mechanisms, we claim that *dynamic learning* is essential for our mechanism to achieve improved performance and the above features.

We consider two widely-used models in this dissertation, a customer bidding model and a posted-price model. For each model, we describe a dynamic learning mechanism with theoretical guarantee of near-optimal performance. Although different models entail different mechanism details and different analysis tools, we show that a common denominator is that the improvement of our mechanisms comes from one unique feature: *dynamic learning*. We also present numerical tests for both models from which the performance of our mechanisms is verified. We believe that the *dynamic learning* mechanisms presented in this dissertation have important implications for both researchers and practitioners.

Acknowledgements

First, I would like to express my sincere gratitude to my advisor, Professor Yinyu Ye. I first met him in the spring of 2006 when he was teaching a short course in Tsinghua University. At that time, I had no idea what operations research was about and it was him who led me into this fascinating field. In the past five years in Stanford, he has continuously advised, encouraged and supported me, both in academic research and other aspects of life. This dissertation would not have been possible without him. Having him as my advisor is one of the greatest fortune I had in my life.

I would like to thank Professor Peter Glynn, for those insightful discussions on several research topics in the past few years. His broad knowledge and inspiring insights have deeply impressed me and I benefited greatly from discussions with him. I would also like to thank Professor Tze Leung Lai, Professor Ben Van Roy and Professor Michael Harrison for serving on my reading and oral committees and providing valuable suggestions and feedbacks on my dissertation.

I would also like to thank my colleagues and collaborators. In the past years, I am very fortunate to have the chance to collaborate with many outstanding colleagues. Among them are Shipra Agrawal, John Gunnar Carlsson, Hongyang Chen, Erick Delage, Shiming Deng, Yichuan Ding, Robert Eberhart, Dongdong Ge, Ming Hu, Mark Peters, Jiawei Zhang and Song Zheng. I have received great helps from them. They are all great colleagues and great friends.

My friends have given me invaluable supports and encouragements in the past years and played an indispensable role in my life in Stanford. My special thanks go to Yichuan Ding, Dongdong Ge, Yuankai Ge, Yihan Guan, Song Li, Xiaoyu Liu, Wenxiu Ma, Chen Peng, Qi Qi, Yu Wu and Xiaowei Zhang. I would also like to thank

my bridge friends who have experienced many excitements with me in this important aspect of my life, and the Association of Chinese Students and Scholars at Stanford (ACSSS) where I spent much enjoyable time and met many of my best friends.

Finally, I would like to express my deepest gratitude to my family. My parents, Xiaopeng Wang and Jun Zhu, have been providing me continuous and unconditional supports for my study and life. Special thanks go to my grandfather, Dashou Zhu, who as a great scientist himself, has devoted much of his time and effort to educating me, not only in science, but also to become a better person. This dissertation is dedicated to my entire family.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

In the past few decades, revenue management played an increasingly important role in many industries ranging from fashion to retail, to travel, and to hospitality. Following rapid growth after the deregulation of the US airline industry in the 1970s, revenue management became a key strategy for airlines as well as other businesses. For example, American Airlines reportedly generated $1.4 billion additional revenue over a three-year period around 1988 due to its development of the Dynamic Inventory Allocation and Maintenance Optimizer (DINAMO) system, the first large-scale revenue management system developed in airline industry. On the other hand, airlines that did not keep pace with the development of revenue management systems, for example, PeopleExpress, failed within a few years [28]. Similar stories across different industries are informing a fast growing body of academic research focusing on revenue management over the last two decades [61, 56].

Given the primary aim of revenue management to sell the right product to the right customer at the right time for the right price, the essence of this discipline is in understanding customers' perception of product value and accurately aligning product prices, placement and availability with each customer segment [28]. In real problems, one of the major challenges faced by revenue managers is the lack of specific knowledge about customer's demand. This is particularly true when a new product or new service is under consideration. This question arises when, for example, a fashion retailer starts to sell a new fashion good, or a game company prepares to release a

new iPhone application, or a search engine wants to allocate search keywords to a set of new advertisers. In such situations, there is little historical data a firm can rely on to make good predictions of future demand and thus decision-making in these situations becomes extremely difficult.

Despite this challenge, most prior literature on revenue management takes demand information as known to the decision maker. This information is exploited to obtain structural insights and derive pricing policies [40, 39]. This literature contributes to our knowledge as they help us to understand the important structures of revenue management problems. Moreover, even when the demand information is known, deriving the optimal pricing policy is not an easy task. However, the ignorance of the demand uncertainty in this literature limits their applicability to real problems.

One method to resolve the uncertainty of demand in revenue management problems is to incorporate demand learning. There are many ways to learn the demand information. One way which is common among industry practitioners is to learn the demand information before the selling season, using sales data for similar products or conducting market surveys. This approach, although proven to be effective in certain occasions, may suffer from the problem of inaccuracy since the learning is not based on the demand information of the current product or current selling season. Further, in these approaches, learning is static, and it is not clear how to take advantage of the increasing sales data as the selling season evolves to refine the learning results and make more accurate decisions.

In this dissertation, we address the aforementioned problem by proposing a class of dynamic learning mechanisms for a broad range of revenue management problems. Instead of estimating consumer demand information before the selling season, our mechanism learns "on the fly". That is, our learning mechanism integrates learning and doing into a concurrent procedure during the selling season. We show that our dynamic learning mechanisms have the following features:

1. *Robustness*, our mechanisms work well for a wide range of underlying demand structures and do not need to know the structure beforehand;

2. *Dynamism*, our mechanisms take advantages of increasing sales data as selling

season moves forward and update the demand information along the process;

3. *Optimality*, our mechanisms achieve near-optimal revenues, as compared to a clairvoyant who has complete knowledge of the demand information beforehand.

By comparing with different learning mechanisms, we claim that *dynamic learning* is essential for our mechanism to achieve improved performance and the above features.

To illustrate our results, we study two revenue management models: a customer bidding model and a posted-price model. Both models have a wide range of applications. We propose dynamic learning mechanisms for these models. We show that, although different models entail different mechanisms details and different analysis tools, the advantage of our mechanism is the consequence of dynamic learning. This important feature enables our mechanisms to achieve near-optimal performance in both models implying important implications for both researchers and practitioners.

Typically, the literature most frequently describes parametric or not approaches to learning mechanisms. In the parametric approach, a parametric functional form of the underlying demand model is assumed and learning is focused on the parameters in such models. In the non-parametric approach, such information is not assumed except for very basic regularity conditions on the underlying model. In prior work, scholars often took a parametric approach. Although a parametric approach indeed simplifies this problem to a certain extent and allows for simpler solutions, it runs the risk of model misspecification. If the parametric form is not well-specified, the achieved revenue could be far away from the optimal one [14, 67, 26]. In light of this, we adopt a non-parametric analysis in this dissertation. This feature makes our analysis robust to demand uncertainty. We further argue that, in an asymptotic regime, using a non-parametric approach for the underlying problems does not impose additional loss compared to the parametric approach (even if it is well-specified). This somewhat surprising result suggests that using a non-parametric approach in these problems is robust while *not* very costly.

In Chapter 2, we study a multi-item customer bidding model in which a sequence of customers approach the seller and request a bundle of products by offering a bid

price. The seller, on observing each customer's bid, makes an irrevocable decision whether to accept or reject the bid without knowing any future demand information. The objective of the seller is to maximize revenue while not violating any inventory limit on each product. This model formulates many online resource allocation and revenue management problems including online combinatorial auctions, online adwords allocations, and online routing and packing problems. In Chapter 2, we describe a dynamic learning mechanism for this class of problems under the assumption of random order of arrival and some mild conditions on the size of inventories. Our mechanism works by dynamically updating a threshold price vector at geometric time intervals, where the threshold prices learned from revealed observations in the previous period are used to determine the sequential decisions in the current period. We show that our dynamic learning mechanism improves upon a "one-time" learning mechanism in which the threshold price is learned only once. By showing a worst-case example, we claim that our mechanism achieves a near-best performance. We present numerical experiments using our mechanism verifying the advantage of dynamic learning over one-time learning and also study some implementation issues of our mechanism and its sensitivity to the input parameters.

In Chapter 3, we consider a posted-price model in which a retailer sells a single product with a potential limited on-hand inventory over a finite selling season. The customers' demands arrive according to a Poisson process, the instantaneous rate of which is influenced by a posted-price controlled by the retailer through an unknown demand function. The seller's decision in this problem is to choose a pricing policy such that his total expected revenue over the entire selling season is maximized. For this problem, we propose a similar dynamic learning mechanism that works robustly across a wide class of underlying demand functions. Our mechanism dynamically updates via price experiments throughout the selling horizon to eventually find the "optimal" selling price. We demonstrate that the performance of our mechanism improves over the previous research's and by providing a worst-case analysis, we show that it actually achieves the best asymptotic performance. Numerical experiments are also presented for this model.

Chapter 2

Dynamic Learning Mechanisms for Customer Bidding Model

2.1 Introduction

Customer bidding model is one of the popular selling models in many businesses. Unlike a posted-price model in which the customers observe a set of listed prices for different products, the customer bidding model allows customers to select a bundle of products of his interest and bid for a certain price for the bundle. The seller, on observing the bundles chosen by the customers as well as the bid prices, decides whether to accept or reject each customer's request with an overall objective of maximizing his revenue while not exceeding the inventory constraints. Such decision problems are sometimes studied under the name of combinatorial auction problems. We refer the readers to [66] for a more thorough review of this subject.

In this chapter, we study such decision problems when the customer's demand is uncertain. That is, instead of observing all the requests all at once, customers arrive sequentially and on observing a request from a certain customer, the decision maker has to make an irrevocable decision whether to accept or reject his request without observing any requests of future customers. Although sometimes unnoticeable, these decisions are made every second in our life. For example, these decisions are faced by search engines who have to assign advertisement slots to advertisers, network

managers who have to allocate bandwidths to customers that request certain network resources, electric grid controllers who want to control the charging times of electric vehicles and airlines that receive a booking request from a customer through bidding websites like Priceline.

In this chapter, we study the above decision problems in a larger framework which we call the "online linear program" model. An online linear program takes linear programming as its underlying form, while the constraint matrix is revealed column by column with the corresponding coefficient in the objective function. After observing the input arrived so far, an online decision must be made regarding the current decision variable without observing the future data. To be precise, consider the following (offline) linear program[1]:

$$\begin{array}{lll} \text{maximize} & \sum_{j=1}^{n} \pi_j x_j & \\ \text{subject to} & \sum_{j=1}^{n} a_{ij} x_j \leq b_i, & i = 1, \ldots, m \\ & 0 \leq x_j \leq 1 & j = 1, \ldots, n \end{array} \tag{2.1}$$

where $\pi_j \geq 0$, $\boldsymbol{a}_j = (a_{ij})_{i=1}^m \in [0,1]^m$, for all j and $\boldsymbol{b} = \{b_i\}_{i=1}^m \in \mathbb{R}^m$.

In the corresponding online problem, instead of observing all the input data at the beginning, the coefficients $(\pi_t, \boldsymbol{a}_t)$ are revealed only at time t, and decision x_t must be made before the next information is revealed, that is, given the previous $t-1$ decisions $x_1, \ldots, x_{t-1}$, and input $\{\pi_j, \boldsymbol{a}_j\}_{j=1}^t$ until time t, one has to make a decision x_t immediately such that

$$\begin{array}{l} \sum_{j=1}^{t} a_{ij} x_j \leq b_i, \quad i = 1, \ldots, m \\ 0 \leq x_t \leq 1. \end{array} \tag{2.2}$$

The goal of an online decision policy is to choose x_t's such that the objective function $\sum_{t=1}^{n} \pi_t x_t$ is maximized.

It can be seen that the decision problems we described earlier can be captured

[1] In many real applications, the decision variables are only allowed to take integer values. However, as we point out in Section 2.4.1, when the problem size is large, the difference between the linear program and the integer program is small. Moreover, as discussed in Section 2.8, our mechanism still holds when integer solution is required. Therefore, for the simplicity of discussion, we will use linear programming as the underlying form throughout this chapter.

by this online linear program model. In those problems, the customers' requests are represented by the columns $\boldsymbol{a}_j$'s and the bid prices are captured by π_j's. The right hand side b_i stands for the inventory limit for each resource. In fact, this online linear program characterizes a very broad class of online decision making problems. We will review several more applications of this model in Section 2.3.

The first question when discussing such a model is what knowledge we assume to have on the input data. For example, an ideal assumption is that we know all the input data at the beginning. In that case, the problem is a static linear program and the optimal decisions are easy to obtain. However, this assumption does not capture the demand uncertainty faced by practical decision makers. One approach which is popular among researchers as well as practitioners, is to assume that the input data is drawn from a certain known distribution. Although a priori input distribution can simplify the problem to a great extent, the choice of distribution is very critical and the performance can suffer if the actual input distribution is not as assumed [16]. Another approach, which is on the other extreme, is to adopt the worst-case analysis. The worst-case analysis, which is completely robust to input uncertainty, evaluates a decision policy based on its performance on the worst-case input [53, 22]. However, this approach is usually too conservative, and leads to very pessimistic theoretical result: no online decision policy can achieve better than $O(1/n)$ fraction of the optimal offline revenue in the worst case [9]. In our work, we take an intermediate path between using a worst-case analysis and assuming the distribution of the input is known. We make the following assumptions on the input model:

Assumption 1 *The columns $\boldsymbol{a}_j$ (with the objective coefficient π_j) arrive in a random order. The set of columns $(\boldsymbol{a}_1, \boldsymbol{a}_2, ..., \boldsymbol{a}_n)$ can be adversarily picked at the start. However, the arrival order of $(\boldsymbol{a}_1, \boldsymbol{a}_2, ..., \boldsymbol{a}_n)$ is uniformly distributed over all the permutations.*

Assumption 1 is strictly weaker than assuming a distributional knowledge of the input. It is satisfied if the input columns are generated independently from some common (but possibly unknown) distributions. In many real problems, the arrival order of customers can be viewed as random in a reasonable length of period and

Assumption 1 is met under such scenarios.

We also make another assumption on the knowledge of the total number of customers.

Assumption 2 *We know the total number of columns n a priori.*

As will be shown later, Assumption 2 is needed since we will use quantity n to decide the length of history used to learn a set of threshold prices in our mechanism. It can be relaxed to an approximate knowledge of n (within $1 \pm \epsilon$ multiplicative error), without affecting our results. In many real situations, decision makers have a good estimate of the arrival rate of the customers, thus the total number of customers can be estimated relatively well. In fact, this assumption is standard in many existing literature for online decision problems [9, 32, 36, 45] and it is necessary for one to obtain a near-optimal performance for any mechanism: In [32], the authors showed that if Assumption 2 does not hold, then the performance of any mechanism is bounded away by a constant ratio from optimal (comparing to the offline optimal revenue).

In the remaining of this chapter, we call the input model satisfying Assumptions 1 and 2 the random permutation model. In the following, we define our criterion to evaluate a certain mechanism.

Definition 1 *Given the set of customers' requests $\mathcal{A} = (\pi, A)$. Let $OPT(\mathcal{A})$ denote the optimal objective value for the offline problem (2.1). We call an online decision policy δ to be c-competitive in random permutation model if the expected revenue obtained by δ is at least c factor of the optimal offline revenue, for any input data described in (2.1). That is,*

$$\mathbb{E}_\sigma \left[\sum_{t=1}^{n} \pi_t x_t^\delta(\sigma, \mathcal{A}) \right] \geq c OPT(\mathcal{A}), \qquad \forall \mathcal{A}$$

where the expectation is taken over uniformly random permutations σ of $1, \ldots, n$, and $x_t^\delta(\sigma, \mathcal{A})$ is the t^{th} decision made by policy δ when the inputs arrive in the order σ.

Note that $\mathrm{OPT}(\mathcal{A})$ is an upper bound for the performance of any decision policy since it represents the revenue achieved by the optimal allocation. The competitiveness of policy δ measures the relative difference between the expected revenue

achieved by using δ and the optimal revenue that could have been obtained if one knows all the demand information. This definition of competitiveness is also adopted in several earlier literature [9, 32, 53, 37, 36].

In this chapter, we also extend our results to the following more general online linear optimization problems:

- Problem (2.2) with both buy and sell orders, that is,

$$\pi_j \text{ can be either positive or negative, and } \boldsymbol{a}_j = (a_{ij})_{i=1}^m \in [-1,1]^m. \tag{2.3}$$

- Problems with multi-dimensional decisions at each time step. More precisely, consider a sequence of n non-negative vectors $\boldsymbol{f}_1, \boldsymbol{f}_2, \ldots, \boldsymbol{f}_n \in \mathbb{R}^k$, mn non-negative vectors

$$\boldsymbol{g}_{i1}, \boldsymbol{g}_{i2}, \ldots, \boldsymbol{g}_{in} \in [0,1]^k, \quad i = 1, \ldots, m,$$

and $(k-1)$-dimensional simplex $K = \{\boldsymbol{x} \in \mathbb{R}^k : \boldsymbol{e}^T\boldsymbol{x} \leq 1, \boldsymbol{x} \geq 0\}$. In this problem, given the previous $t-1$ decisions $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{t-1}$, each time we make a k-dimensional decision $\boldsymbol{x}_t \in \mathbb{R}^k$, satisfying:

$$\begin{array}{c} \sum_{j=1}^t \boldsymbol{g}_{ij}^T \boldsymbol{x}_j \leq b_i, \quad i = 1, \ldots, m \\ \boldsymbol{x}_t \in K \end{array} \tag{2.4}$$

where decision vector $\boldsymbol{x}_t$ must be chosen only using the knowledge up to time t. The objective is to maximize $\sum_{j=1}^n \boldsymbol{f}_j^T \boldsymbol{x}_j$ over the entire time horizon. Note that Problem (2.2) is a special case of Problem (2.4) with $k = 1$.

2.1.1 Key Ideas and Main Results

In our work, we present a near-optimal mechanism for the online linear program (2.2) under the random permutation model. In this section, we present the key ideas of our mechanism and its achieved competitiveness.

Our mechanism is based on the observation that the optimal solution $\boldsymbol{x}^*$ for the

offline linear program (2.1) can be well determined by the optimal dual solution $\boldsymbol{p}^* \in \mathbb{R}^m$ corresponding to its m inequality constraints (except for some degenerate situations). The optimal dual solution acts as a *threshold price* so that $x_j^* > 0$ only if $\pi_j \geq \boldsymbol{p}^{*T}\boldsymbol{a}_j$. Our online mechanism works by learning a threshold price vector from the input received so far. The price vector then determines the allocation decision for the next period. However, instead of computing a new price vector at every arrival of new customer, the mechanism initially waits until ϵn steps or arrivals, and then computes a new price vector every time the history doubles, that is, at time steps $\epsilon n, 2\epsilon n, 4\epsilon n, \ldots$ and so on. We show that our mechanism is $1 - O(\epsilon)$-competitive in random permutation model under a size condition of the right-hand-side input. Our main results are precisely stated as follows:

Theorem 1 *For any $\epsilon > 0$, our online mechanism is $1 - O(\epsilon)$ competitive for the online linear program (2.2) in the random permutation model, for all inputs such that*

$$B = \min_i b_i \geq \Omega\left(\frac{m \log(n/\epsilon)}{\epsilon^2}\right). \tag{2.5}$$

Note that the condition in Theorem 1 depends on $\log n$, which is far from satisfying every customer's demand when n is large. In [45], the author proves that $B \geq 1/\epsilon^2$ is necessary to get $1 - O(\epsilon)$ competitive ratio in k-secretary problem, which is the single dimensional case of our problem with $m = 1$ and $a_t = 1$ for all t. Thus, dependence on ϵ in the above theorem is near-optimal. In the next theorem, we show that a dependence on m is necessary as well for any online mechanism to obtain a near-optimal solution.

Theorem 2 *For any online algorithm for linear program (2.2) in random permutation model, there exists an instance such that its competitive ratio is less than $1-\Omega(\epsilon)$ when*

$$B = \min_i b_i \leq \frac{\log(m)}{\epsilon^2}.$$

We also extend our results to more general online linear programs as introduced in (2.3) and (2.4):

Theorem 3 *For any $\epsilon > 0$, our mechanism is $1 - O(\epsilon)$ competitive for the general online linear problem (2.3) or (2.4) in random permutation model, for all inputs such that:*

$$B = \min_i b_i \geq \Omega\left(\frac{m \log(nk/\epsilon)}{\epsilon^2}\right). \tag{2.6}$$

Remark 1 *Our condition to hold the main result is independent of the size of OPT (see Section 2.8 for an extension of Theorem 1 using OPT in the bound) or objective coefficients, and is also independent of any possible distribution of input data. If the largest entry among constraint coefficients does not equal to* 1*, then both of our theorems hold if condition* (2.5) *and* (2.6) *are replaced by:*

$$\frac{b_i}{\bar{a}_i} \geq \Omega\left(\frac{m \log(nk/\epsilon)}{\epsilon^2}\right), \ \forall i,$$

where for each row i, $\bar{a}_i = \max_j\{|a_{ij}|\}$ for Problem (2.2), or $\bar{a}_i = \max_j\{\|\boldsymbol{g}_{ij}\|_\infty\}$ for Problem (2.4).

It is apparent that our generalized problem formulation should cover a wide range of online decision making problems. In the next two sections, we discuss the related work and some applications of our model. As one will see, our result in fact improves the best competitive ratios for many online problems studied in the literature and is the first non-asymptotic analysis for solving many online resource allocation and revenue management problems.

2.2 Related Work

Online decision making has been a topic of wide recent interest in the computer science, operations research, and management science communities. Various special cases of the general problem presented in this chapter have been studied extensively in the computer science literature as the "secretary problems". A comprehensive survey of existing results on the secretary problems can be found in [9]. In particular, constant factor competitive ratios have been proven for k-secretary and knapsack

secretary problems under the random permutation model[2]. Further, for many of these problems, a constant competitive ratio is known to be optimal if no additional conditions on input are assumed. Therefore, there have been recent interests in searching for online algorithms whose competitive ratio approaches 1 as the input parameters become large. The first result of this kind appears in [45], where a $1 - O(1/\sqrt{k})$-competitive algorithm is presented for k-secretary problem under random permutation model. Recently, [32] and [36] presented a $1-O(\epsilon)$-competitive algorithm for online adwords matching problem under conditions of certain lower bounds on OPT in terms of ϵ and other input parameters. Our work is closely related to the works of [45], [32] and [36]. We compare our contributions to theirs in Table 2.1.

In [45], the author present a recursive algorithm for single dimensional multiple-choice secretary problem, and prove that the $1-O(1/\sqrt{k})$ competitive ratio achieved by his algorithm is the best possible for this problem. Our results extend his work into multi-dimensional cases with competitiveness $(1-O(\sqrt{m \log n/k}))$ $(k = \min_i b_i)$. However, the extension is not trivial due to the multi-dimensional structure of the problem and very different techniques are required for our analysis. We also prove that no online algorithm can achieve a competitive ratio better than $(1 - \Omega(\sqrt{\log m/k}))$ for the multi-dimensional problem. To our knowledge, this is the first result that shows the necessary of dependence on dimension m, for the best competitive ratio achievable for this problem.

In [32], the authors study a one-time pricing approach (as opposed to our dynamic pricing approach) for a special case (adwords allocation problem) of the online linear programming problem. The authors prove a competitive ratio that depends on the optimal offline value OPT, rather than the right hand side $k = \min_i b_i$. To be precise, they prove an algorithm with competitiveness ratio $1 - O(\sqrt[3]{m^2 \log n/\text{OPT}})$. In this work, we use different techniques and develop algorithms to achieve a bound that can depend only on k, and not on the value of OPT. Even in terms of OPT, we derive a competitiveness ratio of $1 - O(\sqrt{m^2 \log n/\text{OPT}})$ for our mechanism (see Section 2.8 and Appendix A.4), which improves over their results. While a competitive ratio in

[2]Here k is the B in our theorems. We use k in the literature review section to make it consistent with the discussions of prior literature.

	Competitiveness
Kleinberg et al. [45]	$1 - \frac{1}{\sqrt{k}}$(only for $m = 1$)
Devanur et al. [32]	$1 - O(\sqrt[3]{m^2 \log n/\mathrm{OPT}})$
Feldman et al. [36]	$1 - O(\max\{\sqrt[3]{m \log n/k}, m \log n/OPT\})$
This chapter [4]	$1 - O(\min\{\sqrt{m \log n/k}, \sqrt{m^2 \log n/OPT}\})$

Table 2.1: Summary and comparison of previous and our results

terms of OPT in general does not imply a corresponding competitive ratio in terms of k and vice-versa (as we illustrate by examples in Appendix A.4), we believe that in practice, having a dependence on k rather than OPT may be more attractive since the value of k is known and can be checked.

More recently, [36] obtain bounds for an online packing problem which depend both on the right hand side k and OPT. In addition to requiring lower bounds on both k and OPT, their lower bound on k depends on ϵ^3. Thus our result is stronger than theirs. Besides, their algorithm is based on a one-time pricing approach as compared to our dynamic pricing algorithm.

In the operations research and management science community, a dynamic and optimal pricing strategy for various online resource allocation problems has always been an important research topic, some literature includes [34, 39, 40, 60, 25, 17]. In [40, 39, 17], the arrival processes are assumed to be price sensitive. However, as commented in [25], this model can be reduced to a price independent arrival process with availability control under Poisson arrivals. Our model can be further viewed as a discrete version of the availability control model which is also used as an underlying model in [60] and discussed in [25]. The idea of using a threshold - or "bid" - price is not new. It is initiated in [68, 58] and investigated further in [60]. In [60], the authors show that the bid price control is asymptotically optimal. However, they assume the knowledge on the arrival process is known and therefore the price is obtained by "forecasting" the future using the distribution information rather than "learning" from the past observations as we do in this chapter. The idea of using linear programming to find dual optimal bid price is discussed in [25] where asymptotic

optimality is achieved. However, [25] also assumes that the arrival process is known.

Our work improves upon the existing works in various manners. First, we provide a near-optimal solution for a wide class of online linear programs which encompasses many special cases of resource allocation and revenue management problems discussed above. Moreover, due to its dynamic learning capability, our mechanism is *distribution free*–no knowledge on the input distribution is assumed except for the random order of arrival. The techniques proposed in this chapter may also be considered a step forward from the threshold price learning approaches. A common element in the techniques used in existing works on secretary problems [9] (with the exception of [45]), online combinatorial auction problems [3], and adwords matching problem [32, 36], has been *one-time learning* of threshold price(s) from first few customers. And the threshold price is then used to determine the decisions for remaining customers. However, in practice one would expect some benefit from dynamically updating the prices as more and more information is revealed. However, the question is: how often and when should one update the price? In [25], the authors demonstrate with a specific example that updating the price too frequently may even hurt the results. In this chapter, we propose a dynamic learning mechanism that updates the prices at geometric time intervals–not too sparse nor too often. In particular, we present an improvement from a factor of $1/\epsilon^3$ to $1/\epsilon^2$ in the lower bound requirement on B by using dynamic price updating instead of one-time learning. Thus we present a precisely quantified strategy for dynamic price update.

In our analysis, we apply many techniques from Probably Approximately Correct (PAC) learning, in particular, concentration bounds and covering arguments. We will point them out as we go into our proofs.

2.3 More Applications

In the following, we show several applications of the online linear programming model. It is worth noting that for many of the problems we discuss below, our mechanism is the first near-optimal mechanism under the distribution-free model.

Online Routing Problems

One of the problems characterized by model (2.1) is the online routing problem, which is an important problem in network management. In this problem, there are m edges in a network, each edge i has a bounded capacity b_i. There are n users arriving online, each with a request for certain path $\boldsymbol{a}_t \in \{0,1\}^m$ and a utility (or bid price) π_t for his request.

By Theorem 1, and the extension to integer programs discussed in Section 2.8, our mechanism gives a $1 - O(\epsilon)$ competitive solution to this problem in the random permutation model as long as the edge capacity satisfies (2.5). Earlier, a best of $\log(m\frac{\pi_{max}}{\pi_{min}})$ competitive algorithm was known for this problem under the worst case input model [21].

Online Adwords Problems

Online adwords allocation problem is a problem that is faced by search engines second by second. In this problem, there are n queries (search keywords) arriving online. There are m bidders (advertisers) each with a daily budget b_i, bidding π_{ij} on query j. For j^{th} query, the decision for the search engine is an m-dimensional vector $\boldsymbol{x}_j$, where $x_{ij} \in \{0,1\}$ indicates whether the j^{th} query is allocated to the i^{th} bidder. Since every query can be allocated to at most one bidder, we have the constraint $\boldsymbol{e}^T\boldsymbol{x}_j \leq 1$. The offline problem can be written as follows:

$$\begin{array}{lll} \text{maximize} & \sum_{j=1}^{n} \boldsymbol{\pi}_j^T \boldsymbol{x}_j & \\ \text{subject to} & \sum_{j=1}^{n} \pi_{ij} x_{ij} \leq b_i, & i = 1, \ldots, m \\ & \boldsymbol{e}^T \boldsymbol{x}_j \leq 1 & j = 1, \ldots, n \\ & \boldsymbol{x}_j \in \{0,1\}^m & j = 1, \ldots, n. \end{array}$$

The linear relaxation of the above problem is a special case of the general linear optimization problem (2.4) with $\boldsymbol{f}_j = \boldsymbol{\pi}_j$, $\boldsymbol{g}_{ij} = \pi_{ij}\boldsymbol{e}_i$ where $\boldsymbol{e}_i$ is the ith unit vector of all zeros except 1 for the ith entry. By Theorem 3, and the extension to integer programs discussed in Section 2.8, our mechanism provides a $1 - O(\epsilon)$ approximation

for this problem given the lower bound condition that for all i, $\frac{b_i}{\pi_i^{max}} \geq \frac{m \log(mn/\epsilon)}{\epsilon^2}$, where $\pi_i^{max} = \max_j\{\pi_{ij}\}$ is the largest bid by bidder i among all queries.

There were several previous studies based on the worst-case analysis on this problem [53, 20, 37] where an algorithm with competitive ratio $1-1/e$ was proposed. This competitive ratio was also proved to be the best ratio that one can achieve. Under the random permutation assumption, [41] showed that the greedy algorithm yields a $1-1/e$ approximation. Later, [38] improved this ratio to 0.67.

Earlier a $1-O(\epsilon)$ algorithm was provided for this problem by [32]. In our work, we eliminate the condition on OPT and only have a condition on b_i which may be checkable before execution, a property which is not provided by the condition of the former type. Further, our lower bounds on B are weaker by an ϵ factor to the one obtained in [32]. We show that this improvement is a result of using dynamic learning, instead of one-time learning. Another near-optimal algorithm for this problem was proposed in [36]. However, their results are also weaker than ours, as shown in the comparison in Table 2.1. Richer models incorporating other aspects of sponsored search such as multiple slots, can be formulated by redefining $\boldsymbol{f}_j, \boldsymbol{g}_{ij}, K$ and similar results can be obtained.

Continuous-time Revenue Management Problems

In our formulation, the customer's demand is assumed to arrive in discrete time slots. In much revenue management literature, such arrivals are assumed to be a stationary Poisson process. To translate a continuous-time problem into ours, we shall discretize the time horizon T into sufficiently small time periods such that there is at most one arrival in each time period and setting $a_t = \mathbf{0}$ for period t if there is no arrival in that period. As discussed in [60], the resulting model will well approximate the arrival process under the assumption of stationary Poisson arrivals. Furthermore, the arrivals can be viewed as randomly ordered. Therefore, the online linear programming model can well handle the continuous-time revenue management problem and our mechanism provides a near-optimal solution to this problem under mild assumptions.

The rest of this chapter is organized as follows. In Section 2.4 and 2.5, we present

our learning mechanisms and prove the competitive ratios under mild conditions on the input. To keep the discussion clear and easy to follow, we start in Section 2.4 with a simpler one-time learning mechanism. While the analysis for this simpler mechanism is useful to demonstrate our proof techniques, the result obtained is weaker than that obtained by our dynamic learning mechanism which is discussed in Section 2.5. In Section 2.6, we give a detailed proof of Theorem 2 regarding the necessity of lower bound condition used in our main theorem. We present results from numerical experiments in Section 2.7 and study some implementation issues of our mechanism. In Section 2.8, we present several extensions, including an alternative bound in terms of the optimal offline value OPT instead of B, the extension to multi-dimensional problems, problems with negative coefficients, integer programs and the applicability of our model to solving large static linear programs. We conclude this chapter in Section 2.9.

2.4 One-time Learning Mechanism

For the linear program (2.1), we use $\boldsymbol{p} \in \mathbb{R}^m$ to denote the dual variable associated with the first set of constraints $\sum_t \boldsymbol{a}_t x_t \leq \boldsymbol{b}$. Let $\hat{\boldsymbol{p}}$ denote the optimal dual solution to the following partial linear program defined only on the input until time $s = \lceil \epsilon n \rceil$ ($\lceil x \rceil$ denotes the smallest integer that is greater than x):

$$\begin{array}{lll} \text{maximize} & \sum_{t=1}^{s} \pi_t x_t & \\ \text{subject to} & \sum_{t=1}^{s} a_{it} x_t \leq (1-\epsilon)\frac{s}{n} b_i, & i = 1, \ldots, m \\ & 0 \leq x_t \leq 1, & t = 1, \ldots, s. \end{array} \tag{2.7}$$

Also, for any given dual price vector $\boldsymbol{p}$, define the allocation rule $x_t(\boldsymbol{p})$ as:

$$x_t(\boldsymbol{p}) = \begin{cases} 0 & \text{if } \pi_t \leq \boldsymbol{p}^T \boldsymbol{a}_t \\ 1 & \text{if } \pi_t > \boldsymbol{p}^T \boldsymbol{a}_t. \end{cases} \tag{2.8}$$

We now state our one-time learning mechanism in Algorithm 1:

In the one-time learning mechanism, we learn a dual price vector using the first

Algorithm 1 One-time Learning Mechanism

1. Initialize $s = \lceil \epsilon n \rceil$, $x_t = 0$, for all $t \leq s$. And $\hat{\boldsymbol{p}}$ is defined as above.
2. Repeat for $t = s+1, s+2, \ldots$
   - If $a_{it} x_t(\hat{\boldsymbol{p}}) \leq b_i - \sum_{j=1}^{t-1} a_{ij} x_j$, set $x_t = x_t(\hat{\boldsymbol{p}})$; otherwise, set $x_t = 0$. Output x_t.

$\lceil \epsilon n \rceil$ arrivals. Then, at each time $t > \lceil \epsilon n \rceil$, it uses this price vector to decide the current allocation, and executes this decision as long as it doesn't violate any of the constraints. An attractive feature of this mechanism is that it requires to solve only one small linear program, defined on $\lceil \epsilon n \rceil$ variables. Note that the right hand side of (2.7) is modified by a factor $1 - \epsilon$. This modification is to guarantee that with high probability, the allocation $x_t(\boldsymbol{p})$ does not violate the constraints. This trick is also used in Section 2.5 when we discuss a dynamic learning mechanism. In the next subsection, we prove the following proposition regarding the competitive ratio of the one-time learning mechanism, which relies on a stronger condition than Theorem 1:

Proposition 1 *For any $\epsilon > 0$, the one-time learning mechanism is $1-6\epsilon$ competitive for the online linear program (2.2) in random permutation model, for all inputs such that*

$$B = \min_i b_i \geq \frac{6m \log(n/\epsilon)}{\epsilon^3}. \tag{2.9}$$

2.4.1 Competitive Ratio Analysis

Observe that the one-time learning mechanism waits until time $s = \lceil \epsilon n \rceil$, and then sets the solution at time t as $x_t(\hat{\boldsymbol{p}})$, unless there is a constraint violation. To prove its competitive ratio, we first prove that with high probability, $x_t(\hat{\boldsymbol{p}})$ satisfies all the constraints of the linear program. Then, we show that the expected revenue $\sum_t \pi_t x_t(\hat{\boldsymbol{p}})$ is close to the optimal offline revenue. For the simplicity of our discussion, we assume $s = \epsilon n$ in the remaining discussions.

To start with, we show that if $\boldsymbol{p}^*$ is the optimal dual solution to (2.1), then $\{x_t(\boldsymbol{p}^*)\}$ is close to the primal optimal solution $\boldsymbol{x}^*$, i.e., learning the dual price is

sufficient to determine a close primal solution. We make the following simplifying technical assumption:

Assumption 3 *The inputs of this problem are in general position, namely for any price vector* $\boldsymbol{p}$*, there can be at most* m *columns such that* $\boldsymbol{p}^T\boldsymbol{a}_t = \pi_t$*.*

The assumption is not necessarily true for all inputs. However, as pointed out by [32], one can always randomly perturb π_t by arbitrarily small amount η through adding a random variable ξ_t taking uniform distribution on interval $[0, \eta]$. In this way, with probability 1, no $\boldsymbol{p}$ can satisfy $m + 1$ equations simultaneously among $\boldsymbol{p}^T\boldsymbol{a}_t = \pi_t$, and the effect of this perturbation on the objective can be made arbitrarily small. Given this assumption, we can use complementary conditions of linear program (2.1) to observe that:

Lemma 1 $x_t(\boldsymbol{p}^*) \leq x_t^*$*, and under Assumption 3,* x_t^* *and* $x_t(\boldsymbol{p}^*)$ *differ at no more than* m *values of* t*.*

Proof of Lemma 1. Let $\boldsymbol{x}^*, \boldsymbol{p}^*$ be the optimal primal-dual solution to the offline problem (2.1). From KKT conditions of (2.1), $x_t^* = 1$, if $(\boldsymbol{p}^*)^T\boldsymbol{a}_t < \pi_t$ and $x_t^* = 0$ if $(\boldsymbol{p}^*)^T\boldsymbol{a}_t > \pi_t$. Therefore, $x_t(\boldsymbol{p}^*) = x_t^*$ if $(\boldsymbol{p}^*)^T\boldsymbol{a}_t \neq \pi_t$. And by Assumption 3, there are at most m values of t such that $(\boldsymbol{p}^*)^T\boldsymbol{a}_t = \pi_t$. □

The lemma tells that, if the optimal dual solution $\boldsymbol{p}^*$ to (2.1) is known, then the (integer) solution $x_t(\boldsymbol{p}^*)$ obtained by an online decision policy is close to the optimal offline solution. However, in an online mechanism, we use the price $\hat{\boldsymbol{p}}$ learned from the first few inputs, instead of the optimal dual price. The remaining discussion attempts to show that the learned price is sufficiently accurate for our purpose. Note that the random order assumption can be interpreted as that the first s inputs are uniform random samples without replacement of size s from the n inputs. Let S denote this sample set of size s, and N denote the complete data of size n. Consider the sample linear program (2.7) defined on the sample set S with right hand side set as $(1-\epsilon)\epsilon\boldsymbol{b}$. Then, $\hat{\boldsymbol{p}}$ was constructed as the optimal dual price of the sample linear program, which we refer as the sample dual price. In the following two lemmas, we show that

with high probability, the primal solution $x_t(\hat{\boldsymbol{p}})$ constructed from this sample dual price is feasible and near-optimal:

Lemma 2 *The primal solution constructed using sample dual price is a feasible solution to the linear program (2.1) with high probability. More precisely, with probability* $1-\epsilon$,

$$\sum_{t=1}^{n} a_{it} x_t(\hat{\boldsymbol{p}}) \leq b_i, \quad \forall i = 1, \ldots, m$$

given $B \geq \frac{6m\log(n/\epsilon)}{\epsilon^3}$.

Proof of Lemma 2: The proof will proceed as follows: Consider any fixed price $\boldsymbol{p}$. We say a random sample S is "bad" for this $\boldsymbol{p}$ if and only if $\boldsymbol{p} = \hat{\boldsymbol{p}}(S)$, but $\sum_{t=1}^{n} a_{it} x_t(\boldsymbol{p}) > b_i$ for some i. First, we show that the probability of bad samples is small for every fixed $\boldsymbol{p}$. Then, we take a union bound over all "distinct" prices to prove that with high probability the learned price $\hat{\boldsymbol{p}}$ will be such that $\sum_{t=1}^{n} a_{it} x_t(\hat{\boldsymbol{p}}) \leq b_i$ for all i.

To start with, we fix $\boldsymbol{p}$ and i. Define $Y_t = a_{it} x_t(\boldsymbol{p})$. If $\boldsymbol{p}$ is an optimal dual solution for sample linear program on S, then by the complementary conditions, we have:

$$\sum_{t\in S} Y_t = \sum_{t\in S} a_{it} x_t(\boldsymbol{p}) \leq (1-\epsilon)\epsilon b_i.$$

Therefore, the probability of bad samples is bounded by:

$$\begin{aligned}
& P(\sum_{t\in S} Y_t \leq (1-\epsilon)\epsilon b_i, \sum_{t\in N} Y_t \geq b_i) \\
\leq\ & P(|\sum_{t\in S} Y_t - \epsilon \sum_{t\in N} Y_t| \geq \epsilon^2 b_i | \sum_{t\in N} Y_t = b_i) \\
\leq\ & 2\exp\left(\frac{-\epsilon^3 b_i}{2+\epsilon}\right) \leq \delta
\end{aligned}$$

where $\delta = \frac{\epsilon}{m\cdot n^m}$. The last step follows from the Hoeffding-Bernstein's Inequality (Lemma 24 in Appendix A.1), and the condition made on B.

Next, we take a union bound over all "distinct" $\boldsymbol{p}$'s. We call two price vectors $\boldsymbol{p}$ and $\boldsymbol{q}$ distinct if and only if they result in distinct solutions, i.e., $\{x_t(\boldsymbol{p})\} \neq \{x_t(\boldsymbol{q})\}$.

Note that we only need to consider distinct prices, since otherwise all the Y_t's are exactly the same. Thus, each distinct $\boldsymbol{p}$ is characterized by a unique separation of n points $(\{\pi_t, \boldsymbol{a}_t\}_{t=1}^n)$ in m-dimensional space by a hyperplane. By results from computational geometry [54], the total number of such distinct prices is at most n^m. Taking union bound over n^m distinct prices, and $i = 1, \ldots, m$, we get the desired result. □

Above we showed that with high probability, $x_t(\hat{\boldsymbol{p}})$ is a feasible solution. In the following, we show that it actually is a near-optimal solution.

Lemma 3 *The primal solution constructed using sample dual price is a near-optimal solution to the linear program* (2.1) *with high probability. More precisely, with probability* $1 - \epsilon$*,*

$$\sum_{t \in N} \pi_t x_t(\hat{\boldsymbol{p}}) \geq (1 - 3\epsilon) OPT \tag{2.10}$$

given $B \geq \frac{6m \log(n/\epsilon)}{\epsilon^3}$.

Proof of Lemma 3: The proof of this lemma is based on two observations. First, $\{x_t(\hat{\boldsymbol{p}})\}_{t=1}^n$ and $\hat{\boldsymbol{p}}$ satisfies all the complementarity conditions, and hence is an optimal primal-dual solution to the following linear program

$$\begin{array}{lll} \text{maximize} & \sum_{t \in N} \pi_t x_t & \\ \text{subject to} & \sum_{t \in N} a_{it} x_t \leq \hat{b}_i, & i = 1, \ldots, m \\ & 0 \leq x_t \leq 1, & t = 1, \ldots, n \end{array} \tag{2.11}$$

where $\hat{b}_i = \sum_{t \in N} a_{it} x_t(\hat{\boldsymbol{p}})$ if $\hat{p}_i > 0$, and $\hat{b}_i = \max\{\sum_{t \in N} a_{it} x_t(\hat{\boldsymbol{p}}), b_i\}$, if $\hat{p}_i = 0$.

Second, we show that if $\hat{p}_i > 0$, then with probability $1 - \epsilon$, $\hat{b}_i \geq (1 - 3\epsilon) b_i$. Let $\hat{\boldsymbol{p}}$ be the optimal dual solution of the sample linear program on set S and $\hat{x}$ be the optimal primal solution. By the complementarity conditions of the linear program, if $\hat{p}_i > 0$, the i^{th} constraint must be satisfied by equality. That is, $\sum_{t \in S} a_{it} \hat{x}_t = (1 - \epsilon)\epsilon b_i$. Then, given the observation made in Lemma 1, and that $B = \min_i b_i \geq \frac{m}{\epsilon^2}$, we get:

$$\sum_{t \in S} a_{it} x_t(\hat{\boldsymbol{p}}) \geq \sum_{t \in S} a_{it} \hat{x}_t - m \geq (1 - 2\epsilon)\epsilon b_i. \tag{2.12}$$

Then, using the Hoeffding-Bernstein's Inequality, in a manner similar to the proof of Lemma 2, we can show that (the proof is given in Appendix A.1.) given the lower bound on B, with probability at least $1-\epsilon$:

$$\hat{b}_i = \sum_{t \in N} a_{it} x_t(\hat{\boldsymbol{p}}) \geq (1-3\epsilon) b_i. \tag{2.13}$$

Lastly, observing that whenever (2.13) holds, given an optimal solution x^* to (2.1), $(1-3\epsilon)x^*$ will be a feasible solution to (2.11). Therefore, the optimal value of (2.11) is at least $(1-3\epsilon)$OPT, which is equivalently saying that

$$\sum_{t=1}^{n} \pi_t x_t(\hat{\boldsymbol{p}}) \geq (1-3\epsilon)\text{OPT}.$$

□

Therefore, the objective value for online solution taken over the entire period $\{1,\ldots,n\}$ is near optimal. However, our mechanism does not make any decision in the learning period S, and only the decisions from period $\{s+1,\ldots,n\}$ contribute to the objective value. The following lemma that relates sample optimal to the optimal value of a linear program will bound the contribution from the learning period:

Lemma 4 *Let OPT(S) denote the optimal value of the linear program* (2.7) *over sample S, and OPT(N) denote the optimal value of the offline linear program* (2.1) *over N. Then,*

$$\mathbb{E}[OPT(S)] \leq \epsilon OPT(N).$$

Proof of Lemma 4: Let $(\boldsymbol{x}^*, \boldsymbol{p}^*, \boldsymbol{y}^*)$ and $(\hat{\boldsymbol{x}}, \hat{\boldsymbol{p}}, \hat{\boldsymbol{y}})$ denote the optimal primal-dual solution of linear program (2.1) on N, and sample linear program on S, respectively.

$$(\boldsymbol{p}^*, \boldsymbol{y}^*) = \begin{array}{lll} \arg\min & \boldsymbol{b}^T\boldsymbol{p} + \sum_{t\in N} y_t & \\ \text{s.t.} & \boldsymbol{p}^T\boldsymbol{a}_t + y_t \geq \pi_t & t \in N \\ & \boldsymbol{p}, \boldsymbol{y} \geq 0 & \end{array} \qquad (\hat{\boldsymbol{p}}, \hat{\boldsymbol{y}}) = \begin{array}{lll} \arg\min & (1-\epsilon)\epsilon\boldsymbol{b}^T\boldsymbol{p} + \sum_{t\in S} y_t & \\ \text{s.t.} & \boldsymbol{p}^T\boldsymbol{a}_t + y_t \geq \pi_t, & t \in S \\ & \boldsymbol{p}, \boldsymbol{y} \geq 0 & \end{array}$$

Since $S \subseteq N$, $(\boldsymbol{p}^*, \boldsymbol{y}^*)$ is a feasible solution to the dual of linear program on S, by

weak duality theorem:

$$\mathrm{OPT}(S) \leq \epsilon \boldsymbol{b}^T \boldsymbol{p}^* + \sum_{t \in S} y_t^*.$$

Therefore,

$$\mathbb{E}[\mathrm{OPT}(S)] \leq \epsilon \boldsymbol{b}^T \boldsymbol{p}^* + \mathbb{E}[\sum_{t \in S} y_t^*] = \epsilon(\boldsymbol{b}^T \boldsymbol{p}^* + \sum_{t \in N} y_t^*) = \epsilon \mathrm{OPT}(N).$$

□

Now, we are ready to prove Proposition 1:

Proof of Proposition 1: Using Lemma 2 and Lemma 3, with probability at least $1 - 2\epsilon$, the following event happen:

$$\sum_{t=1}^{n} a_{it} x_t(\hat{\boldsymbol{p}}) \leq b_i, \quad i = 1, \ldots, m$$

$$\sum_{t=1}^{n} \pi_t x_t(\hat{\boldsymbol{p}}) \geq (1 - 3\epsilon) OPT.$$

That is, the decisions $x_t(\hat{\boldsymbol{p}})$ are feasible and the objective value taken over the entire period $\{1, \ldots, n\}$ is near optimal. Denote this event by $\mathcal{E}$, where $P(\mathcal{E}) \geq 1 - 2\epsilon$. We have by Lemma 2, 3 and 4:

$$\begin{aligned}
\mathbb{E}\left[\sum_{t \in N \setminus S} \pi_t x_t(\hat{\boldsymbol{p}}) | \mathcal{E}\right] &\geq (1 - 3\epsilon)\mathrm{OPT} - \mathbb{E}\left[\sum_{t \in S} \pi_t x_t(\hat{\boldsymbol{p}}) \mid \mathcal{E}\right] \\
&\geq (1 - 3\epsilon)\mathrm{OPT} - \frac{1}{1 - 2\epsilon} \mathbb{E}\left[\sum_{t \in S} \pi_t x_t(\hat{\boldsymbol{p}})\right] \\
&\geq (1 - 3\epsilon)\mathrm{OPT} - \frac{\epsilon}{1 - 2\epsilon} \mathrm{OPT}.
\end{aligned}$$

Therefore,

$$\mathbb{E}\left[\sum_{t=s+1}^{n} \pi_t x_t(\hat{\boldsymbol{p}})\right] \geq \mathbb{E}\left[\sum_{t\in N\setminus S} \pi_t x_t(\hat{\boldsymbol{p}}) \mid \mathcal{E}\right] \cdot P(\mathcal{E}) \geq (1-6\epsilon)\mathrm{OPT}. \quad \square \tag{2.14}$$

2.5 Dynamic Learning Mechanism

The mechanism discussed in the previous section uses the first ϵn inputs to learn a threshold price, and then applies it in the remaining time horizon. While this mechanism has its own merits, in particular, requires solving only a small linear problem defined on ϵn variables, the lower bound required on B is stronger than that claimed in Theorem 1 by an ϵ factor.

In this section, we discuss an improved dynamic learning mechanism that will achieve the result in Theorem 1. Instead of computing the price only once, this dynamic learning mechanism will update the price every time the history doubles, that is, it learns a new price at time $t = \epsilon n, 2\epsilon n, 4\epsilon n, \ldots$. To be precise, let $\hat{\boldsymbol{p}}^{\ell}$ denote the optimal dual solution for the following partial linear program defined on the input until time ℓ:

$$\begin{array}{lll} \text{maximize} & \sum_{t=1}^{\ell} \pi_t x_t & \\ \text{subject to} & \sum_{t=1}^{\ell} a_{it} x_t \leq (1-h_\ell)\frac{\ell}{n} b_i, & i = 1,\ldots,m \\ & 0 \leq x_t \leq 1, & t = 1,\ldots,\ell \end{array} \tag{2.15}$$

where the set of numbers h_ℓ are defined as follows:

$$h_\ell = \epsilon\sqrt{\tfrac{n}{\ell}}. \tag{2.16}$$

Also, for any given dual price vector $\boldsymbol{p}$, define the allocation rule $x_t(\boldsymbol{p})$ as earlier in (2.8). Then, our dynamic learning mechanism can be stated as follows:

Note that we update the price $\lceil \log_2(1/\epsilon) \rceil$ times during the whole time horizon. Thus, the mechanism requires more computation, but as we show next it requires a weaker lower bound on B for proving the same competitive ratio. The intuition behind this improvement is as follows. Note that initially, at $\ell = \epsilon n$, $h_\ell = \sqrt{\epsilon} > \epsilon$. Thus,

Algorithm 2 Dynamic Learning Mechanism

1. Initialize $t_0 = \lceil \epsilon n \rceil$, $x_t = \hat{x}_t = 0$, for all $t \leq t_0$.
2. Repeat for $t = t_0 + 1, t_0 + 2, \ldots$
   (a) Set $\hat{x}_t = x_t(\hat{\boldsymbol{p}}^\ell)$, where $\ell = \lceil 2^r \epsilon n \rceil$ for the largest integer r such that $\ell < t$.
   (b) If $a_{it}\hat{x}_t \leq b_i - \sum_{j=1}^{t-1} a_{ij}x_j$, set $x_t = \hat{x}_t$; otherwise, set $x_t = 0$. Output x_t.

more slack is available, and the large deviation argument for constraint satisfaction (as in Lemma 2) requires a weaker condition on B. The numbers h_ℓ decreases as ℓ increases. However, for large values of ℓ, sample size is larger, making the weaker condition on B sufficient for our purpose. Also, h_ℓ decreases rapidly enough, so that the overall loss on the objective value is not significant. The careful choice of numbers h_ℓ will play an important role in proving our results.

2.5.1 Competitive Ratio Analysis

The analysis for the dynamic learning mechanism proceeds in a manner similar to that for the one-time learning mechanism. However, stronger results for the price learned in each period need to be proved here. In this proof, for simplicity of discussion, we assume that $\epsilon = 2^{-E}$ for some integer E, and that the numbers $\ell = 2^r \epsilon n$ for $r = 0, 1, 2, ..., E-1$ are all integers. Let $L = \{\epsilon n, 2\epsilon n, \ldots, 2^{E-1}\epsilon\}$.

Lemma 5 and 6 are parallel to Lemma 2 and 3, in the previous section, however require a weaker condition on B:

Lemma 5 *For any $\epsilon > 0$, with probability $1 - \epsilon$:*

$$\sum_{t=\ell+1}^{2\ell} a_{it}x_t(\hat{\boldsymbol{p}}^\ell) \leq \frac{\ell}{n}b_i, \qquad \textit{for all } i \in \{1, \ldots, m\},\ \ell \in L$$

given $B = \min_i b_i \geq \frac{10m\log(n/\epsilon)}{\epsilon^2}$.

Proof of Lemma 5: The proof is similar to the proof of Lemma 2 but a more careful analysis is needed. We provide a brief outline here with a detailed proof in

Appendix A.2. First, we fix $\boldsymbol{p}$, i and ℓ. This time, we say a random order is "bad" for this $\boldsymbol{p}$, i and ℓ if and only if $\boldsymbol{p} = \hat{\boldsymbol{p}}^l$ but $\sum_{t=\ell+1}^{2\ell} a_{it} x_t(\hat{\boldsymbol{p}}^l) > \frac{l}{n} b_i$. By using the Hoeffding-Bernstein's Inequality, we show that the probability of "bad" orders is less than $\delta = \frac{\epsilon}{m \cdot n^m \cdot E}$ for any fixed $\boldsymbol{p}$, i and ℓ under the condition on B. Then by taking a union bound over all distinct prices, all items i and periods ℓ, the lemma is proved. □

In the following, we introduce some notations. Let $\texttt{LP}_s(\boldsymbol{d})$ denote the partial linear program that is defined on variables till time s with right hand side in the inequality constraints set as $\boldsymbol{d}$. That is,

$$\begin{array}{lll} & \text{maximize} & \sum_{t=1}^{s} \pi_t x_t \\ \texttt{LP}_s(\boldsymbol{d}): & \text{subject to} & \sum_{t=1}^{s} a_{it} x_t \le d_i, \quad i = 1, \ldots, m \\ & & 0 \le x_t \le 1, \qquad\quad t = 1, \ldots, s. \end{array}$$

And let $\text{OPT}_s(\boldsymbol{d})$ denote the optimal objective value for $\texttt{LP}_s(\boldsymbol{d})$.

Lemma 6 *With probability at least $1 - \epsilon$, for all $\ell \in L$:*

$$\sum_{t=1}^{2\ell} \pi_t x_t(\hat{\boldsymbol{p}}^\ell) \ge (1 - 2h_\ell - \epsilon) OPT_{2\ell}(\frac{2\ell}{n}\boldsymbol{b})$$

given $B = \min_i b_i \ge \frac{10m \log(n/\epsilon)}{\epsilon^2}$.

Proof of Lemma 6: Let $\hat{b}_i = \sum_{j=1}^{2\ell} a_{ij} x_j(\hat{\boldsymbol{p}}^\ell)$ for i such that $p_i^\ell > 0$, and $\hat{b}_i = \max\{\sum_{j=1}^{2\ell} a_{ij} x_j(\hat{\boldsymbol{p}}^\ell), \frac{2\ell}{n} b_i\}$, otherwise. Then, the solution pair $(\{x_t(\hat{\boldsymbol{p}}^\ell)\}_{t=1}^{2\ell}, \hat{\boldsymbol{p}}^\ell)$, satisfies all the complementarity conditions, therefore is an optimal primal-dual solution for the linear program $\texttt{LP}_{2\ell}(\hat{\boldsymbol{b}})$:

$$\begin{array}{lll} \text{maximize} & \sum_{t=1}^{2\ell} \pi_t x_t & \\ \text{subject to} & \sum_{t=1}^{2\ell} a_{it} x_t \le \hat{b}_i, \quad i = 1, \ldots, m & (2.17) \\ & 0 \le x_t \le 1, \qquad\quad t = 1, \ldots, 2\ell. & \end{array}$$

This means

$$\sum_{t=1}^{2\ell} \pi_t x_t(\hat{\boldsymbol{p}}^\ell) = \mathrm{OPT}_{2\ell}(\hat{\boldsymbol{b}}) \geq \left(\min_i \frac{\hat{b}_i}{b_i \frac{2\ell}{n}} \right) \mathrm{OPT}_{2\ell}(\frac{2\ell}{n}\boldsymbol{b}). \tag{2.18}$$

Now, let us analyze the ratio $\frac{\hat{b}_i}{2\ell b_i/n}$. By definition, for i such that $p_i^\ell = 0$, $\hat{b}_i \geq 2\ell b_i/n$. Otherwise, using techniques similar to the proof of Lemma 5, we can prove that with probability $1-\epsilon$,

$$\hat{b}_i = \sum_{t=1}^{2\ell} a_{it} x_t(\hat{\boldsymbol{p}}^\ell) \geq (1 - 2h_\ell - \epsilon)\frac{2\ell}{n} b_i. \tag{2.19}$$

A detailed proof of inequality (2.19) appears in Appendix A.2. And the lemma follows from (2.19). □

Next, similar to Lemma 4 in previous section, we prove the following lemma relating the sample optimal to the optimal value of the offline linear program:

Lemma 7 *For any ℓ,*

$$\mathbb{E}\left[OPT_\ell(\frac{\ell}{n}\boldsymbol{b}) \right] \leq \frac{\ell}{n} OPT.$$

The proof of lemma 7 is exactly the same as the proof for Lemma 4.

Now we are ready to prove Theorem 1.

Proof of Theorem 1: Observe that the output of the online solution at time $t \in \{\ell+1, \ldots, 2\ell\}$ is $x_t(\hat{\boldsymbol{p}}^\ell)$ as long as the constraints are not violated. By Lemma 5 and Lemma 6, with probability at least $1-2\epsilon$:

$$\sum_{t=\ell+1}^{2\ell} a_{it} x_t(\hat{\boldsymbol{p}}^\ell) \leq \frac{\ell}{n} b_i, \qquad \text{for all } i \in \{1, \ldots, m\},\ \ell \in L$$

$$\sum_{t=1}^{2\ell} \pi_t x_t(\hat{\boldsymbol{p}}^\ell) \geq (1 - 2h_\ell - \epsilon)\mathrm{OPT}_{2\ell}(\frac{2\ell}{n}\boldsymbol{b}).$$

Denote this event by $\mathcal{E}$, where $P(\mathcal{E}) \geq 1-2\epsilon$. Given this event, the expected

objective value achieved by the online algorithm can be bounded as follows:

$$\begin{aligned}
& \mathbb{E}\left[\sum_{\ell\in L}\sum_{t=\ell+1}^{2\ell}\pi_t x_t(\hat{\boldsymbol{p}}^\ell)|\mathcal{E}\right] \\
\geq\ & \sum_{l\in L}\mathbb{E}\left[\sum_{t=1}^{2\ell}\pi_t x_t(\hat{\boldsymbol{p}}^l)|\mathcal{E}\right]-\sum_{\ell\in L}\mathbb{E}\left[\sum_{t=1}^{\ell}\pi_t x_t(\hat{\boldsymbol{p}}^\ell)|\mathcal{E}\right] \\
\geq\ & \sum_{\ell\in L}(1-2h_l-\epsilon)\mathbb{E}\left[\mathrm{OPT}_{2\ell}(\frac{2\ell}{n}\boldsymbol{b})|\mathcal{E}\right]-\sum_{\ell\in L}\mathbb{E}\left[\mathrm{OPT}_{\ell}(\frac{\ell}{n}\boldsymbol{b})|\mathcal{E}\right] \\
\geq\ & \mathrm{OPT}-\sum_{\ell\in L}2h_l\mathbb{E}\left[\mathrm{OPT}_{2\ell}(\frac{2\ell}{n}\boldsymbol{b})|\mathcal{E}\right]-\epsilon\sum_{\ell\in L}\mathbb{E}\left[\mathrm{OPT}_{2\ell}(\frac{2\ell}{n}\boldsymbol{b})|\mathcal{E}\right]-\mathbb{E}\left[\mathrm{OPT}_{\epsilon n}(\epsilon\boldsymbol{b})|\mathcal{E}\right] \\
\geq\ & \mathrm{OPT}-\frac{1}{P(\mathcal{E})}\left\{\sum_{\ell\in L}2h_l\mathbb{E}\left[\mathrm{OPT}_{2\ell}(\frac{2\ell}{n}\boldsymbol{b})\right]-\epsilon\sum_{\ell\in L}\mathbb{E}\left[\mathrm{OPT}_{2\ell}(\frac{2\ell}{n}\boldsymbol{b})\right]-\mathbb{E}\left[\mathrm{OPT}_{\epsilon n}(\epsilon\boldsymbol{b})\right]\right\} \\
\geq\ & \mathrm{OPT}-\frac{4}{1-2\epsilon}\sum_{\ell\in L}h_l\frac{l}{n}\mathrm{OPT}-\frac{2\epsilon}{1-2\epsilon}\sum_{l\in L}\frac{l}{n}\mathrm{OPT}-\frac{\epsilon}{1-2\epsilon}\mathrm{OPT} \\
\geq\ & \mathrm{OPT}-\frac{13\epsilon}{1-2\epsilon}\mathrm{OPT}
\end{aligned}$$

where the last inequality follows from the fact that

$$\sum_{\ell\in L}\frac{\ell}{n}=(1-\epsilon),\ \text{and}\ \sum_{\ell\in L}h_\ell\frac{\ell}{n}=\epsilon\sum_{\ell\in L}\sqrt{\frac{\ell}{n}}\leq 2.5\epsilon.$$

Therefore,

$$\mathbb{E}\left[\sum_{\ell\in L}\sum_{t=\ell+1}^{2\ell}\pi_t x_t(\hat{\boldsymbol{p}}^\ell)\right]\geq\mathbb{E}\left[\sum_{\ell\in L}\sum_{t=\ell+1}^{2\ell}\pi_t x_t(\hat{\boldsymbol{p}}^\ell)|\mathcal{E}\right]P(\mathcal{E})\geq(1-15\epsilon)\mathrm{OPT}. \tag{2.20}$$

Thus, Theorem 1 is proved. □

2.6 Worst-case Bound for any Mechanism

In this section, we prove Theorem 2, i.e., the condition $B \geq \Omega(\log m/\epsilon^2)$ is necessary for any online mechanism to achieve a competitive ratio of $1-O(\epsilon)$. We prove this by constructing an instance of (2.1) with m items and B units of each item such that no online mechanism can achieve a competitive ratio of $1-O(\epsilon)$ unless $B \geq \Omega(\log m/\epsilon^2)$.

In this construction, we refer to the $0-1$ vectors $\boldsymbol{a}_t$'s as demand vectors, and π_t's as profit coefficients. Assume $m = 2^z$ for some integer z. We will construct z pairs of demand vectors such that demand vectors in each pair are complement of each other, and do not share any item.

However, every set of z vectors consisting of exactly one vector from each pair will share at least one common item. To achieve this, consider the 2^z possible boolean strings of length z. The j^{th} boolean string represents j^{th} item for $j = 1, \ldots, m = 2^z$ (for illustrative purpose, we index the item from 0 in our later discussion). Let s_{ij} denote the value at i^{th} bit of the j^{th} string. Then, construct a pair of demand vectors $\boldsymbol{v}_i, \boldsymbol{w}_i \in \{0,1\}^m$, by setting $v_{ij} = s_{ij}$, $w_{ij} = 1 - s_{ij}$.

Figure 2.1 illustrates this construction for $m = 8$ ($z = 3$):

Demand vectors

Items	v_3	v_2	v_1
0	0	0	0
1	0	0	1
2	0	1	0
3	0	1	1
4	1	0	0
5	1	0	1
6	1	1	0
7	1	1	1

Demand vectors

Items	w_3	w_2	w_1
0	1	1	1
1	1	1	0
2	1	0	1
3	1	0	0
4	0	1	1
5	0	1	0
6	0	0	1
7	0	0	0

Figure 2.1: Illustration of the worst-case bound

Note that pair of vectors $\boldsymbol{v}_i, \boldsymbol{w}_i, i = 1, \ldots, z$ are complement of each other. Consider any set of z demand vectors formed by picking exactly one of the two vectors $\boldsymbol{v}_i$ and $\boldsymbol{w}_i$ for each $i = 1, \ldots, z$. Then form a bit string by setting $s_{ij} = 1$ if this set has vector $\boldsymbol{v}_i$ and 0 if it has vector $\boldsymbol{w}_i$. Then, all the vectors in this set share the

item corresponding to the boolean string. For example, in above, the demand vectors $\boldsymbol{v}_3, \boldsymbol{w}_2, \boldsymbol{w}_1$ share item $4(=' 100')$, the demand vectors $\boldsymbol{w}_3, \boldsymbol{v}_2, \boldsymbol{v}_1$ share item $3(=' 011')$ and so on.

Now, we construct an instance consisting of

- B/z inputs with profit coefficient 4 and demand vector $\boldsymbol{v}_i$, for each $i = 1, \ldots, z$.
- q_i inputs with profit equals to 3 and demand vector $\boldsymbol{w_i}$, for each i, where q_i is a random variable following Binomial$(2B/z, 1/2)$.
- $\sqrt{B/4z}$ inputs with profit 2 and demand vector $\boldsymbol{w_i}$, for each i.
- $2B/z - q_i$ inputs with profit 1 and demand vector $\boldsymbol{w}_i$, for each i.

Using the properties of demand vectors ensured in the construction, we prove the following claim:

Claim 1 *Let r_i denote number of vectors of type $\boldsymbol{w}_i$ accepted by any $1-\epsilon$ competitive solution for the constructed example. Then, it must hold that*

$$\sum_i |r_i - B/z| \leq 7\epsilon B.$$

Proof of Claim 1 Let OPT denote the optimal value of the offline program. And let OPT_i denote the revenue obtained from demands accepted of type i. Let $\text{topw}_i(k)$ denote sum of profits of top k inputs with demand vector $\boldsymbol{w}_i$. Then

$$\text{OPT} = \sum_{i=1}^{z} \text{OPT}_i \geq \sum_{i=1}^{z} (4B/z + \text{topw}_i(B/z)) = 4B + \sum_{i=1}^{z} \text{topw}_i(B/z).$$

Let $\widehat{\text{OPT}}$ be the objective value of a solution which accepts r_i vectors of type $\boldsymbol{w}_i$. Firstly, note that $\sum_i r_i \leq B$. This is because all $\boldsymbol{w}_i$s share one common item, and there are at most B units of this item available. Let Y be the set $\{i : r_i > B/z\}$, and X be the remaining i's, i.e. $X = \{i : r_i \leq B/z\}$. Then, we show that total number of accepted $\boldsymbol{v}_i$s cannot be more than $B - \sum_{i \in Y} r_i + |Y|B/z$. Obviously, the set Y cannot contribute more than $|Y|B/z$ $\boldsymbol{v}_i$s. Let $S \subseteq X$ contribute the remaining

v_is. Now consider the item that is common to all $\boldsymbol{w}_i$s in set Y and $\boldsymbol{v}_i$s in the set S (there is at least one such item by construction). Since only B units of this item are available, total number of v_is contributed by S cannot be more than $B - \sum_{i\in Y} r_i$. Therefore number of accepted $\boldsymbol{v}_i$s is less than or equal to $B - \sum_{i\in Y} r_i + |Y|B/z$.

Denote $P = \sum_{i\in Y} r_i - |Y|B/z$, $M = |X|B/z - \sum_{i\in X} r_i$. Then, $P, M \geq 0$. And the objective value

$$\begin{aligned} \widehat{\text{OPT}} &\leq \textstyle\sum_{i=1}^{z} \text{topw}_i(r_i) + 4(B - \sum_{i\in Y} r_i + |Y|B/z) \\ &\leq \textstyle\sum_i \text{topw}_i(B/z) + 3P - M + 4(B - P) \\ &= \text{OPT} - P - M. \end{aligned} \tag{2.21}$$

Since $\text{OPT} \leq 7B$, this means that, $P + M$ must be less than $7\epsilon B$ in order to get an approximation ratio of $1 - \epsilon$ or better. □

Here is a brief description of the remaining proof. By construction, for every i, there are exactly $2B/z$ demand vectors $\boldsymbol{w}_i$ that have profit coefficients 1 and 3, and those have equal probability to take value 1 or 3. Now, from the previous claim, in order to get a near-optimal solution, one must select close to B/z demand vectors of type $\boldsymbol{w}_i$. Therefore, if total number of $(3, \boldsymbol{w}_i)$ inputs are more than B/z, then selecting any $(2, \boldsymbol{w}_i)$ will cause a loss of 1 in profit as compared to the optimal profit; and if total number of $(3, \boldsymbol{w}_i)$ inputs are less than $B/z - \sqrt{B/4z}$, then rejecting any $(2, \boldsymbol{w}_i)$ will cause a loss of 1 in profit. Using central limit theorem, at any step, both these events can happen with constant probability. Thus, every decision for $(2, \boldsymbol{w}_i)$ is a mistake with constant probability, which results in a total expected loss of $\Omega(\sqrt{B/z})$ for every i, that is, a total loss of $\Omega(\sqrt{zB})$.

If the number of $\boldsymbol{w}_i$s to be accepted is not exactly B/z, some of these $\sqrt{B/z}$ decisions may not be mistakes, but as in the claim above, such cases cannot be more than $7\epsilon B$. Therefore, the expected value of online solution,

$$\text{ONLINE} \leq \text{OPT} - \Omega(\sqrt{zB} - 7\epsilon B).$$

Since OPT $\leq 7B$, in order to get $(1-\epsilon)$ approximation factor, we need

$$\Omega(\sqrt{z/B} - 7\epsilon) \leq 7\epsilon \Rightarrow B \geq \Omega(z/\epsilon^2) = \Omega(\log(m)/\epsilon^2).$$

This completes the proof of Theorem 2. A detailed exposition of steps used in this proof appears in Appendix A.3.

2.7 Numerical Results

We present some numerical tests of our learning mechanisms studied in this section. We want to achieve three goals in the numerical analysis: First, we want to study some implementation issues of our dynamic learning mechanism; Second, we want to study the performance of our mechanism and compare it to the performance of the one-time learning mechanism; And last, we want to investigate the relationship between the performance of our mechanism and the input parameters.

We introduce a base problem for our numerical study. In the base problem, there are $m = 5$ products for sale, each with $b_i = 1000$ copies. There are $n = 10000$ customers, the probability of each customer requesting each product is set to be $q = 0.5$[3] (and we remove those empty requests). The bidding price π_j is determined in the following way. We assume there is an underlying price p^* drawn from $U[0,1]^5$, where $U[a,b]$ denotes a uniform random variable on interval $[a,b]$. Then, customer j's bid price π_j will consist of two parts, one is ${p^*}^T A_j$ which can be viewed as the inherent value of bundle j, and the other is a noise term following a standard Gaussian distribution. Although this structure is somewhat restrictive, it reflects in some degree how customers value the products: as in the classical choice model, the utility of a certain product (or a bundle of products) can be decomposed into an observable parts (which is usually the value of the product) and an unobservable part which is usually modeled by a random variable. Finally, we introduce the notion of Relative Loss (RL)

[3]This is a simplifying assumption. However, as we find out in our numerical tests, our result is quite robust in the choice of the columns and therefore we choose this for convenience.

defined as

$$\text{RL} = 1 - \frac{\text{Expected Revenue}}{\text{Offline Optimal Revenue}} \tag{2.22}$$

to evaluate our mechanism in the numerical tests. Note that RL is simply 1 minus the competitive ratio used in our theoretic analysis.

To perform the numerical experiments, one key parameter that has to be determined is the "ϵ", which governs "the time to learn" in our mechanisms (both for one-time learning and dynamic learning). In Theorem 1, we showed that for problem with input parameters m, n and B, the relative loss of the dynamic learning mechanism is approximately of order

$$\sqrt{\frac{m \log n}{B}}.$$

Therefore, a choice of $\epsilon = \sqrt{\frac{m \log n}{B}}$ may be reasonable (correspondingly, choosing $\epsilon = \sqrt[3]{\frac{m \log n}{B}}$ may be reasonable for the one-time learning mechanism). However, this may not be the optimal choice in practice. In particular, in the dynamic learning mechanism, choosing a smaller ϵ may have two benefits. First, it reduces the loss due to the ignorance of the first ϵn customers; second, it increases the number of price updates which intuitively helps the decision maker to refine the learned price and achieve better performance. In the one-time learning mechanism, the choice of ϵ is more subtle. On one hand, a smaller ϵ will reduce the loss due to the ignorance of the first ϵn customers, however, on the other hand, too small ϵ may lead to a less accurate dual price $\hat{p}$ which may result in a poor revenue. In the following, we test our learning mechanisms on the base problem for different choices of ϵ. We generate one instance of the base problem (one set of A_j's and π_j's), then for each choice of ϵ, we test the performances for both one-time and dynamic learning mechanism for 500 different arrival orders (randomly generated). We then compare the average revenue collected in the 500 arrival orders to the optimal offline revenue. The results are summarized in Table 2.2 and Figure 2.2.

In Table 2.2, we show the average RL for both mechanisms as well as the standard

ϵ	RL of One-time Learning(Std)	RL of Dynamic Learning(Std)
0.01	13.38%(6.38%)	2.96%(1.61%)
0.02	11.93%(5.09%)	3.38%(1.67%)
0.03	9.31%(4.47%)	3.58%(1.41%)
0.04	9.04%(3.68%)	4.83%(2.01%)
0.05	9.22%(4.23%)	5.20%(2.23%)
0.06	9.23%(3.55%)	5.98%(1.96%)
0.07	9.36%(3.91%)	7.08%(2.45%)
0.10	9.99%(3.24%)	9.53%(2.19%)
0.15	14.85%(3.52%)	14.85%(2.69%)
0.20	19.76%(3.05%)	19.68%(2.27%)

Table 2.2: Performance comparison between dynamic learning and one-time learning mechanisms for different choices of ϵ

deviations for different values of ϵ's. We observe:

- For the dynamic learning mechanism, choosing a small ϵ improves its performance. This suggests that in practical implementations, one should choose a small ϵ for this mechanism. As we discussed earlier, this improvement comes from two factors, one is the decrease of the loss due to the ignorance of the initial customers, the other is the more price updating involved in the mechanism. Moreover, as we can see from Table 2.2 and Figure 2.2, the relative loss scales with ϵ approximately linearly when ϵ is larger than 0.06, however, this is not the case when ϵ is small. This observation is in accordance with Theorem 1, which says that only when $B \geq O(\frac{m \log(n/\epsilon)}{\epsilon^2})$, the loss is of order ϵ.

- For the one-time learning mechanism, the optimal choice of ϵ is more subtle. If ϵ is too small, then there will not be enough data to learn, while if ϵ is too large, the cost of ignoring the first ϵn customer is high. Therefore an ϵ that is of medium range is optimal. However, the exact optimal choice of ϵ will depend on the problem specification (asymptotically, one should choose $\epsilon \sim O(\sqrt[3]{\frac{m \log n}{B}})$,

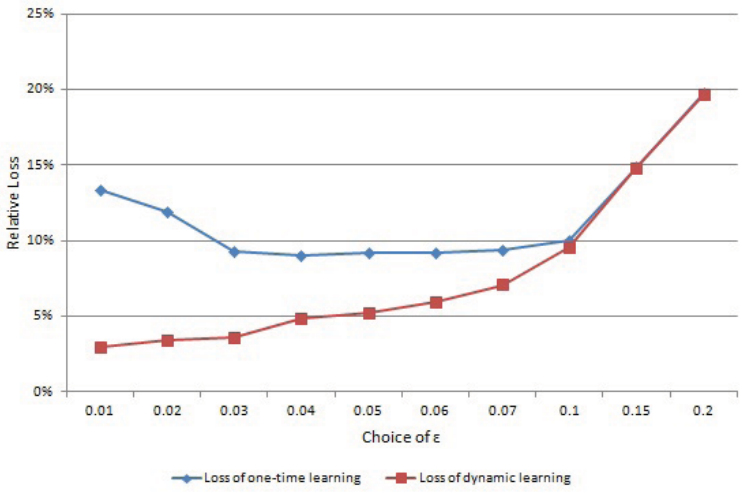


Figure 2.2: Performance comparison between dynamic learning and one-time learning mechanisms for different choices of ϵ

but the optimal constant is not clear). Therefore, even for the convenience of choosing ϵ, dynamic learning mechanism has its advantage.

- From Table 2.2 and Figure 2.2, we can see that the dynamic learning mechanism outperforms one-time learning for all the choices of ϵ: it doesn't only have a smaller average RL, but also has a smaller variance. However, the difference between these two mechanisms becomes less significant if a large ϵ is used: in that case, the main loss is due to the ignorance of the first ϵn customers, rather than the accuracy of the learned price.

Here we remark on the run time of both mechanisms. One drawback of the dynamic learning mechanism is that it requires the decision maker to solve more and larger-sized linear programs. However, given the maturity of LP solvers, this is usually not a big problem. In the numerical examples we considered above, it takes about 0.7 second on a small laptop to run the dynamic learning mechanism once (with $n = 10000$) , comparing to 0.1 second for the one-time learning mechanism (we use the MATLAB LP solver). This speed is usually satisfiable in real problems.

Now we have shown that the dynamic learning mechanism indeed achieves better performance than the one-time learning mechanism in nearly all the cases and we

m	1	2	3	4	5
Relative Loss	1.42%	2.38%	2.74%	2.91%	2.93%
m	6	8	10	15	20
Relative Loss	3.29%	3.49%	3.86%	4.09%	4.39%

Table 2.3: Relative loss of the dynamic learning mechanism vs. the number of products m

should always choose a smaller ϵ for the dynamic learning mechanism. In the remaining of this section, we will focus on the dynamic learning mechanism and further study how each input parameter affects its performance. We will set $\epsilon = 0.01$ for all our remaining experiments.

First, we study how m (the number of products) affects the performance of our mechanism. We consider the base problem with different values of m's ranging from 1 to 20. For each m, we generate 20 sets of requests (A's and π's), and for each set of requests, we generate 20 random arrival orders. To evaluate the performance, we first take an average of the performance of the 20 arrival orders, and then further take an average across the 20 sets of requests. This effectively takes an average of the relative loss for all the 400 instances in the numerical experiment. The result is summarized in Table 2.3.

In Table 2.3, we observe that the relative loss of our dynamic learning mechanism increases as m increases, which verifies the dependence on m in the performance bound. More interestingly, we find that the relative loss has a very good linear relationship with $\sqrt{\log m}$. For this, we performed linear regression (without constant term) for the relative loss with respect to $\sqrt{\log m}$ and found that the goodness of fit (R-square) achieves 0.983. This goodness of fit is much larger than that if we perform linear regression to m ($R^2 = 0.691$), $\sqrt{m}$ ($R^2 = 0.874$) or $\log m$ ($R^2 = 0.948$). This relationship is illustrated in Figure 2.3. Note that in Theorem 2, we prove that the worst-case relative loss for this problem is of order $\sqrt{\log m}$. Therefore, our numerical results suggest that 1) The worst-case bound in Theorem 2 is likely to be tight and 2) Our dynamic learning mechanism has already achieved the best possible performance

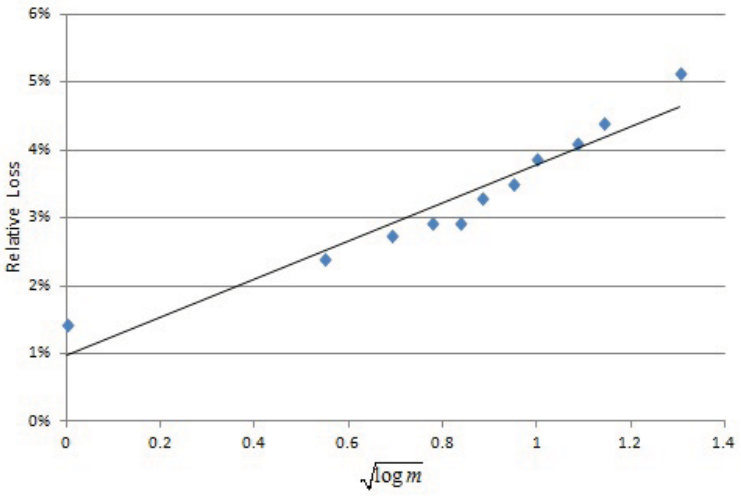


Figure 2.3: Relative loss of the dynamic learning mechanism vs. the number of products m

B	250	500	750	1000	1500
Relative Loss	7.82%	5.94%	4.95%	2.96%	2.59%
B	2000	2500	3000	4000	5000
Relative Loss	2.34%	2.16%	1.35%	1.24%	1.12%

Table 2.4: Relative loss of the dynamic learning mechanism vs. the inventory level B

for this problem.

Next, we test the performance of our mechanism for different values of B's. We vary B in the base problem from 250 to 5000. Note that since the probability of certain customer requesting certain product is set to 0.5, $B = 5000$ roughly means that everybody's request can be met. Again, for each B, we generate 20 sets of requests, and for each set of requests, we generate 20 random order of arrivals. We then take an average of the relative loss for these 400 instances. The result is summarized in Table 2.4.

In Table 2.4, we can see that the relative loss of our mechanism decreases as B increases. This is in accordance to Theorem 1 and also intuitive: as there are more units of product to sell, the loss of learning should be smaller. We try to study the

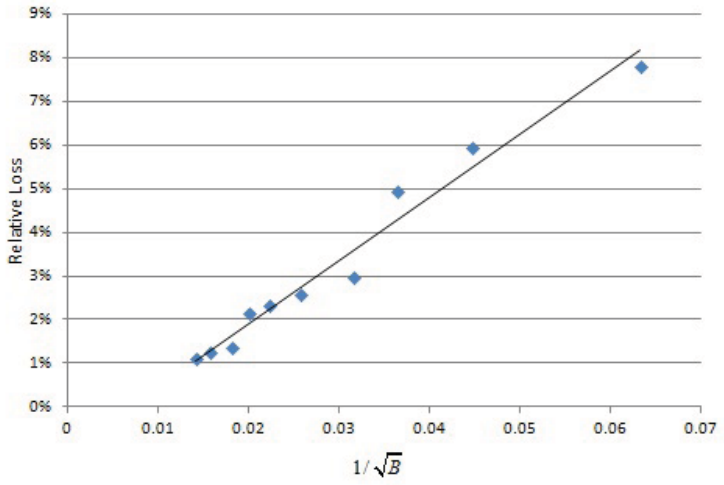


Figure 2.4: Relative loss of the dynamic learning mechanism vs. the inventory level B

n	2500	5000	7500	10000	15000
Relative Loss	2.93%	3.19%	3.2%	2.96%	2.99%
n	20000	25000	30000	40000	50000
Relative Loss	2.93%	2.74%	2.50%	2.42%	2.36%

Table 2.5: Relative loss of the dynamic learning mechanism vs. the total number of customers n

relationship between the relative loss and B and find that the relative loss has a linear relationship with $1/\sqrt{B}$. The goodness of fit is 0.978 comparing to that of 0.916 if we fit the relative loss to B. This relationship is shown in Figure 2.4 and is in accordance with both Theorem 1 and Theorem 2.

Finally, we test the performance of our mechanism for different values of n's. We vary n in the base problem from 2500 to 50000 and use the same testing method. The result is shown in Table 2.5.

In Table 2.5, we see that the relative regret slightly decreases as n increases. However, in Theorem 1, the relative regret is increasing with n (in the order of $\sqrt{\log n}$). Our conjecture is that we can drop the n in this bound, however, we do not

	Theorem 1	Worst-Case Bound	Numerical Result
Relative Loss vs. B	$O(\sqrt{1/B})$	$O(\sqrt{1/B})$	$O(\sqrt{1/B})$
Relative Loss vs. m	$O(\sqrt{m})$	$O(\sqrt{\log m})$	$O(\sqrt{\log m})$
Relative Loss vs. n	$O(\sqrt{\log n})$	Independent	Slightly decreasing in n

Table 2.6: Summary of numerical experiments

yet know how to achieve this. This is an interesting future direction of research.

To summarize, in this section, we show that our dynamic learning mechanism indeed performs better than the one-time learning mechanism in test problems. We also observe interesting relationships between the relative loss of the dynamic learning mechanism and the input parameters as summarized in Table 2.6. These observations provide much information for future research.

2.8 Extensions

Alternative Bounds in terms of OPT

In Theorem 1, we required lower bound conditions on the right-hand side input $B = \min_i b_i$. Interestingly, our conditions can be easily translated into a corresponding condition that depends only on the optimal offline objective value OPT instead of B. We have the following proposition.

Proposition 2 *For any $\epsilon > 0$, our dynamic learning mechanism is $1 - O(\epsilon)$ competitive for the online linear program (2.2) in random permutation model, for all inputs such that*

$$\frac{OPT}{\pi_{\max}} \geq \Omega\left(\frac{m^2 \log(n/\epsilon)}{\epsilon^2}\right) \tag{2.23}$$

where $\pi_{\max} = \max_{j=1,\ldots,n} \pi_j$.

The preferable form of condition may depend on the application at hand. However, we remark that the above proposition even improves upon the existing results that depend on OPT ([32], [36]) by reducing the dependence on ϵ from ϵ^3 to ϵ^2. A key

to these improvements is our dynamic learning approach as opposed to the previous one-time learning approach. Meanwhile, we can show that none of the two forms of conditions (on OPT as in Proposition 2, and on B as in Theorem 1) dominates the other. In Appendix A.4, we give examples showing that either of the conditions could be stronger depending on the problem instance.

General Constraint Matrices

In (2.1), we assumed that each entry of the constraint matrix is between zero and one and the coefficients in the objective function is positive. Here, we show that these restrictions can be relaxed to

$$\pi_j \text{ either positive or negative, and } \quad \boldsymbol{a}_j = (a_{ij})_{i=1}^m \in [-1,1]^m.$$

First it is easy to note that the requirement $\pi_j \geq 0$ can be relaxed since our analysis didn't depend on the positiveness of π. When negative entries exist in the constraint matrix, everything in the proof is the same except that there is risk of violating the constraint (2.2) at certain step of the mechanism. The following lemma which is a strengthened statement of Lemma 5 shows that with high probability this will not happen in our dynamic learning mechanism.

Lemma 8 *For any $\epsilon > 0$, with probability $1-\epsilon$:*

$$\sum_{t=\ell+1}^{w} a_{it} x_t(\hat{\boldsymbol{p}}^\ell) \leq \frac{\ell}{n} b_i, \qquad \textit{for all } i \in \{1,\ldots,m\},\ \ell \in L,\ \ell+1 \leq w \leq 2\ell$$

given $B = \min_i b_i \geq \frac{10(m+1)\log(n/\epsilon)}{\epsilon^2}$.

Proof of Lemma 8: The idea of the proof is to add an extra n factor to the δ defined in proving Lemma 5 to guarantee that for every step, our mechanism will not violate the inventory constraint. This will only affect the condition on B by increasing m to $m+1$ which is no more than a constant factor. The detailed proof is given in Appendix A.5. □

Note that allowing negative a_{ij}'s means that we not only allow to sell products, but also allow buying from others. This may have important implications in many management problems.

Online Multi-dimensional Linear Program

We consider the following more general online linear programs with multi-dimensional decisions $\boldsymbol{x}_t \in \mathbb{R}^k$ at each step, as defined in Section 2.1:

$$\begin{array}{lll} \text{maximize} & \sum_{t=1}^n \boldsymbol{f}_t^T \boldsymbol{x}_t & \\ \text{subject to} & \sum_{t=1}^n \boldsymbol{g}_{it}^T \boldsymbol{x}_t \leq b_i & i = 1, \ldots, m \\ & \boldsymbol{e}^T \boldsymbol{x}_t \leq 1, \boldsymbol{x}_t \geq 0 & \forall t \\ & \boldsymbol{x}_t \in \mathbb{R}^k. & \end{array} \tag{2.24}$$

Our learning mechanism remains essentially the same (as described in Section 2.5), with $\boldsymbol{x}_t(\boldsymbol{p})$ now defined as follows:

$$\boldsymbol{x}_t(\boldsymbol{p}) = \begin{cases} 0 & \text{if for all } j,\ f_{tj} \leq \sum_i p_i g_{itj} \\ \boldsymbol{e}_r & \text{otherwise, where } r \in \arg\max_j (f_{tj} - \sum_i p_i g_{itj}). \end{cases} \tag{2.25}$$

Using the complementary conditions of (2.24), and the lower bound condition on B as assumed in Theorem 3, we can prove the following lemmas; the proofs are very similar to the proofs for the one-dimensional case, and will be provided in Appendix A.4.

Lemma 9 *$\boldsymbol{x}_t^*$ and $\boldsymbol{x}_t(\boldsymbol{p}^*)$ differ at no more than m values of t.*

Lemma 10 *We call $\boldsymbol{p}$ and $\boldsymbol{q}$ are distinct if $\boldsymbol{x}_t(\boldsymbol{p}) \neq \boldsymbol{x}_t(\boldsymbol{q})$ for some t. Then, there are at most $n^m k^{2m}$ distinct price vectors.*

With the above lemmas, the proof of Theorem 3 will follow exactly as the proof for Theorem 1.

Online Integer Programs

From the definition of $x_t(\boldsymbol{p})$ in (2.8), our mechanisms always output integer solutions. And, since the competitive ratio analysis will compare the online solution to the optimal solution of the corresponding linear relaxation, the competitive ratio stated in Theorem 1 also holds for the online integer programs. The same observation holds for the general online linear program introduced in Section 2.8 since it also outputs integer solutions. Our result implies a common belief: when relatively sufficient resource quantities are to be allocated to a large number of small demands, linear programming solutions possess a small gap to integer programming solutions.

Fast Solution of Large Linear Programs by Column Sampling

Apart from online problems, our mechanism can also be applied for solving (offline) linear programs that are too large to consider all the variables explicitly. Similar to the idea of one-time learning mechanism, one could randomly sample a small subset ϵn of variables, and use the dual solution $\hat{\boldsymbol{p}}$ for this smaller program to set the values of variables x_j as $x_j(\hat{\boldsymbol{p}})$. This approach is very similar to the popular column generation method used for solving large linear programs [29]. Our result provides the first rigorous analysis of the approximation achieved by the approach of reducing the linear program size by randomly selecting a subset of columns.

2.9 Conclusions

In this chapter, we provided a near-optimal mechanism for a general class of online linear programming problems under the assumption of random order of arrival and some mild conditions on the right-hand-side input. The conditions we used are independent of the optimal objective value, objective function coefficients, or distributions of input data.

Our dynamic learning mechanism works by dynamically updating a threshold price vector at geometric time intervals. Our result provides the best known upper and lower bounds for many generalized secretary problems including online routing,

adwords allocation problems, etc; also it presents a first non-asymptotic analysis for the online LP, a model frequently used in revenue management and resource allocation problems.

There are many questions for future research. One important question is whether the current bound on the size of the right-hand-input B is tight? Currently as we show in this paper, we have a gap between our mechanism and the lower bound. Filling that gap would be a very interesting direction for future research.

Chapter 3

Dynamic Learning Mechanisms for Posted-price Model

3.1 Introduction

In this chapter, we study the pricing decisions in another selling model: the posted-price model. In such a model, a seller decides and adjusts the posted-prices for the products he wants to sell and the arriving customers make their purchase decisions based on the price posted by the seller. The products usually have certain availability constraints and the objective of the seller is to find a pricing policy to maximize his revenue. The use of such posted-price selling model is widespread across the business world.

We focus on a simple case of this problem when there is only a single product for sale and study the above decision problem when the underlying demand model of the customer is unknown. More precisely, we assume that the customer arrives (and makes purchases) in a Poisson process, the instantaneous rate of which is influenced only by the posted-price controlled by the seller through a demand function. However, the demand function is unknown and the information regarding it can only be obtained through observing realized demand given certain posted-price. Our goal is to design a learning mechanism such that the seller could learn the demand function "on the fly" with the overall objective of maximizing his total expected revenue in

the entire selling season.

There are several reasons why we are interested in this problem. Most existing research in revenue management assumes that the functional relationship between demand and retailer's action is known to the decision maker. This relationship is then exploited to derive pricing policies. However, in reality, decision makers seldom possess such information. This is especially true when a new product or service is introduced or the market environment has changed. For example, when a fashion retailer starts to sell a new fashion good, or a game company prepares to release a new iPhone application, there is usually very little historical data they can rely on to make accurate demand forecasts. Efforts to obtain this information before the selling season (e.g. performing market surveys or test sales) may frequently be inaccurate because of the bias inherent in those approaches. Therefore, people have been looking for alternative solutions in which learning is accomplished during the selling season. A "learning-while-doing" mechanism has the potential to overcome the problems stated above, but the key questions remain as to how the learning and doing can be accomplished together and how well we can achieve.

There are two types of learning models that one can use to approach this problem: parametric and non-parametric. In parametric approaches, one assumes that prior information has been obtained about which parametric family the demand function belongs to. Decision makers then take actions "on the fly" while updating their beliefs about the underlying parameters. On the other hand, in non-parametric approaches, one does not assume any structural properties of the demand function except some basic regularity conditions. And it is the decision maker's task to learn the demand curve with very limited information. Although a parametric approach indeed helps to simplify the problem to a great extent, it suffers from one major drawback of running the risk of model misspecification. In real problems, one may frequently encounter situations when the demand function does not behave so nicely that it satisfies any parametric functional form. If one insists on using a parametric approach, the result could be very costly. It has been shown in several literature [14, 16] (also shown in our numerical experiments in Section 3.7) that if the parametric model is not properly specified, the revenue achieved could be much less than the optimal revenue that

could have been achieved.

In light of the above reasons, we take a non-parametric approach in this chapter. Instead of assuming the underlying demand function takes a certain parametric functional form, we only make very limited regularity assumptions on the demand functions (see Assumption A in Section 3.3) and study a learning mechanism that would work robustly. We then study the value of knowing the parametric information of the demand function (if it exists) by comparing the performance of our mechanism and the optimal revenue when that information is known. This value will be important for decision makers to decide how much they should invest in collecting the information of parametric form of the demand function before the selling season.

3.1.1 Our Contributions

Our main contribution in this chapter is to propose a dynamic learning mechanism for the single product posted-price revenue management problem stated above. We show that the performance of our mechanism improves over several previous proposed mechanisms for this problem in terms of the asymptomatic regret (which is the relative loss to the optimal revenue when the demand function is known, and as problem size grows large). Furthermore, by showing a worst-case bound, we prove that our mechanism provides the near best asymptotic performance over all possible pricing policies. We show that this gain of the performance in our mechanism is due to the use of "dynamic learning" rather than "one-time learning", that is, our mechanism does not separate learning and doing into two phases, instead, we combine them in a concurrent procedure in which our mechanism iteratively performs price experimentations within a series of shrinking price intervals. This feature of our mechanism itself might be of interest to revenue management researchers and practitioners.

More details of our results and some implications are summarized as below:

1. Under mild regularity conditions on the demand function, we show that our pricing mechanism achieves an asymptotic regret of $O(n^{-1/2})$ (with some logarithmic factor), uniformly across all possible demand functions. This result improves the previously best-known result by Besbes and Zeevi [14] for both

the non-parametric learning (achieves an asymptotic regret of order $O\left(n^{-1/4}\right)$) and the parametric learning (achieves an asymptotic regret of order $O\left(n^{-1/3}\right)$ when a general parametric model is concerned) and closes the efficiency gap between them. Our result implies that there is no additional cost associated with performing non-parametric learning, in terms of asymptotic regret.

2. Our result also shows that the two selling models, customer-bidding model and posted-price model, achieve the same performance in terms of asymptotic revenue. In Chapter 2, we discuss a dynamic learning mechanism with $O(n^{-1/2})$ regret under the customer bidding mechanism (if we assume the inventory of each good is of order n and equate the model into a continuous time framework). We obtain the same result in this chapter for the posted-price model, although in a posted-price model the seller only learns a threshold information for each customer's valuation versus a complete information in the customer bidding model. Therefore, our result provides an reassurance for the usage of the posted-price mechanism, which is more widespread in practice.

The rest of this chapter is organized as follows: In Section 3.2, we review the related literature in this field. In Section 3.3, we introduce our model and state our main assumptions. In Section 3.4, we present our dynamic learning mechanism and the main theorems. We provide a sketch proof for our mechanism in Section 3.5. In Section 3.6, we show a lower bound of regret for all possible pricing policies. We show some numerical results in Section 3.7 and some extensions of this model in Section 3.8. We conclude this chapter in Section 3.9. All detailed proofs for our technical results are deferred to the appendix.

3.2 Literature Review

Pricing mechanisms have been an important research area in revenue management and there is abundant literature on this topic. For a comprehensive review on this subject, we refer the readers to Bitran and Caldentey [17], Elmaghraby and Keskinocak [34],

McGill and van Ryzin [52], and the recent books by Talluri and van Ryzin [61] and Phillips [56]. Previously, research has mostly focused on the cases in which the functional relationship between the price and demand (also known as the demand function) is given to the decision maker. For example, Gallego and van Ryzin [40] present a foundational work in such a setting where the structure of the demand function is exploited and the optimal pricing policy is analyzed.

Although knowing the exact demand function is convenient for analysis, the decision makers in practice do not usually own such information. Therefore, much recent literature addresses the dynamic pricing problems under demand uncertainty. The majority of these work takes a parametric approach, e.g., Lobo and Boyd [49], Bertsimas and Perakis [13], Araman and Caldentey [5], Aviv and Pazgal [7], Carvalho and Puterman [23], Farias and Van Roy [35], Broder and Rusmevichientong [19] and Harrison et al [42]. Typically in these work, a prior knowledge of the parameters is assumed and a dynamic program with Bayesian updating of the parameters is formulated. Although such approach simplifies the problem to some degree, the restriction to a certain demand function family may incur model misspecification risk. As shown in Besbes and Zeevi [14], misspecifying the demand family may lead to revenues that are far away from the optimum. In such cases, a non-parametric approach would be preferred since it does not commit to any family of demand function.

The main difficulty facing the non-parametric approach is its tractability and efficiency. And most research revolves around this idea. Several studies consider a model in which the customers are chosen adversarially, e.g. Ball and Queyranne [12] and Perakis and Roels [55]. However, their models take a relatively conservative approach and no learning is involved. Rusmevichientong et al [57] consider static learning using historic data with no dynamic decision being made. In another paper by Lim and Shanthikumar [47], they consider dynamic pricing strategies that are robust to an adversarial at every point in the stochastic process. Again, this approach is quite conservative and the main theme is about robustness rather than demand learning.

The work that is closest to the discussion in this chapter is that of Besbes and

Zeevi [14] where the authors consider demand learning in both parametric and non-parametric case. They propose learning mechanisms for both cases and show that there is a performance gap between them. They also provide a lower bound for the revenue loss in both cases. In this chapter, we improve their results for both cases and close the performance gap. In particular, they consider algorithms with separated learning and doing phases where price experimentation is performed exclusively during the learning phase (except the parametric case with a single parameter). In our work, the learning and doing is dynamically integrated: we keep shrinking a price interval that contains an "optimal price" and keep learning until we guarantee that the revenue achieved by applying the current price is near-optimal. This idea of dynamic learning distinguishes our work from theirs and makes our results stronger.

Another paper that is related to ours is Kleinberg and Leighton [45]. In [45], the authors consider the online posted-price auction with unlimited supply. They show lower bounds for the revenue loss for three cases: 1) each customer has a same but unknown valuation, 2) each customer has an i.i.d. unknown valuation, and 3) the valuations of customers are chosen adversarily to the algorithm. They also provide algorithms that match these three lower bounds. Specifically, for the case where each customer has an i.i.d. valuation, they present an algorithm with the same order of regrets as ours. However, their model is different from ours in several ways. First, they consider an unconstrained revenue maximization problem (without inventory constraint) while we consider a constrained problem (with an inventory constraint). Second, they consider a discrete-time arrival model while we consider a continuous-time Poisson arrival model. As we will see in our algorithm, these differences are nontrivial and our analysis is fundamentally different from theirs.

A recent work by Besbes and Zeevi [16] makes a further study on the risk of model misspecification when a parametric model is used. They show that when a model with a single parameter is used, the risk of model misspecification may indeed be significant. However, surprisingly, they show that when a parametric family with two or more parameters is used (e.g., a linear demand with two parameters), despite the presence of model misspecification, pricing decisions are near-optimal (with asymptotic regret of order $O(n^{-1/2})$). This result is remarkable as it justifies

the use of a parametric approach in many situations. However, one drawback in their result is that they make an assumption on the underlying demand function ($(1/2)\lambda(p)|\lambda''(p)|/(\lambda'(p))^2 < 1$ for all p) which is not necessary for our mechanism (and is not satisfied by certain demand functions that may occur in practice). Also, they don't allow an inventory constraint in their model and they consider a discrete time model as oppose to the continuous time model studied in this chapter.

Other related literature that focuses on the exploration-exploitation trade-off in sequential optimization under uncertainty is from the study of the multi-armed bandit problem: see Lai and Robbins [46], Agrawal [2] and Auer et al [6] and references therein. Although our study shares similarity in ideas with the multi-armed bandit problem, we consider a problem with continuous learning horizon (the time is continuous), continuous learning space (the possible demand function is continuous) and continuous action space (the price set is continuous). These features in addition to the presence of inventory constraint make our mechanism and analysis unique.

3.3 Model

In this chapter, we consider the problem of a single seller selling a single product in a finite selling season T. The seller has a fixed inventory x at the beginning and no recourse actions on the inventory can be made during the selling season. Here, the inventory constraint is optional as we allow x to take the value of ∞. The decision for the seller is to post a price $p(t)$ at each moment during the selling season, where $p(t)$ can only depend on all the prices and demand realizations before time t. The demand of this product arrives according to a Poisson process with instantaneous demand rate at time t being λ_t. In our model, we assume that λ_t is solely determined by $p(t)$, that is, we can write $\lambda_t = \lambda(p(t))$ as a function of $p(t)$. At time T, the sales will be terminated and there is no salvage cost for the remaining items (as shown in [40], the zero salvage cost assumption is without loss of generality). The objective is to maximize the expected revenue of the seller over the entire selling season.

We assume the feasible set of prices is an interval $[\underline{p}, \overline{p}]$ with an addition cut-off price p_∞ such that $\lambda(p_\infty) = 0$. Regarding the demand rate function $\lambda(p)$, we assume

it is decreasing in p, has an inverse function $p = \gamma(\lambda)$, and the revenue rate function $r(\lambda) = \lambda\gamma(\lambda)$ is concave in λ. These assumptions are quite standard and such demand functions bear the name of "regular" demand function as defined in [40].

Besides being regular, we also make the following assumptions on the demand rate function $\lambda(p)$ and the revenue rate function $r(\lambda)$:

Assumption A. For some positive constants M, K, m_L and m_U,

1. Boundedness: $|\lambda(p)| \leq M$ for all $p \in [\underline{p}, \overline{p}]$;
2. Lipschitz continuity: $\lambda(p)$ and $r(\lambda(p))$ are Lipschitz continuous with respect to p with factor K. Also, the inverse demand function $p = \gamma(\lambda)$ is Lipschitz continuous in λ with factor K;
3. Strict concavity and differentiability: $r''(\lambda)$ exists and $-m_L \leq r''(\lambda) \leq -m_U < 0$ for all λ in the range of $\lambda(p)$ for $p \in [\underline{p}, \overline{p}]$.

In the following, let $\Gamma = \Gamma(M, K, m_L, m_U)$ denote the set of demand functions satisfying the above assumptions with the corresponding coefficients. We briefly illustrate these assumptions as follows: The first assumption is simply an upper bound on the demand rate. The second assumption says that when we change the price by a small amount, the demand and revenue rate will not change by too much, also the demand function does not have a "flat" part. These two assumptions are quite standard as they appear in most literature in revenue management with demand learning, e.g., [40], [14] and [19]. The last assumption contains two parts, one being the smoothness of the revenue function, the other being strict concavity. Note that the assumption on the existence of second derivatives is also made by [19, 45, 16], and the strict concavity assumption is made by [45, 16]. And as shown in Appendix B.1, our assumptions hold for several classes of commonly-used demand functions (e.g., linear, exponential and multi-nomial logit demand functions).

In our model, we assume that the seller does not know the true demand function λ, except that it belongs to Γ. Note that Γ doesn't need to have any parametric representation. Therefore, our model is robust in terms of the choice of the demand

function family.

In the following, we define the criterion to evaluate the performance of any pricing mechanism. Consider a pricing policy π. At each time t, π maps all the history prices and realized demand information into a current price $p(t)$. By our assumption that the demand follows a Poisson process, the cumulative demand up to time t can be written as:

$$N^{\pi}(t) = N\left(\int_0^t \lambda(p(s))ds\right), \tag{3.1}$$

where $N(\cdot)$ is a unit-rate Poisson process. In order to satisfy the inventory constraint, any admissible policy π must satisfy:

$$\begin{aligned} &\int_0^T dN^{\pi}(s) \leq x \\ &p(s) \in [\underline{p}, \overline{p}] \cup p_{\infty} \quad 0 \leq s \leq T. \end{aligned} \tag{3.2}$$

We denote the set of policy satisfying (3.2) by $\mathcal{P}$. Note that the seller can always set the price to p_{∞}, thus constraint (3.2) can always be met. And the expected revenue generated by a policy π is given as follows:

$$J^{\pi}(x, T; \lambda) = E\left[\int_0^T p(s)dN^{\pi}(s)\right]. \tag{3.3}$$

Here, the presence of λ in (3.3) means that the expectation is taken under the demand function λ. Given a demand function λ, we wish to find the optimal policy π^* that maximizes the expected revenue (3.3) while subjected to the inventory constraint (3.2). In our model, since we don't have perfect information on λ, we seek a pricing policy π that performs as close to π^* as possible.

Even if the demand function λ is known, computing the expected value of the optimal policy is hard. It involves solving a Bellman equation resulting from a dynamic program. Fortunately, as shown in previous literature [14, 40], we can obtain an upper bound for the expected value for any policy via considering a full-information deterministic optimization problem. Define:

$$\begin{aligned} J^D(x,T;\lambda) = \quad & \sup \quad \int_0^T r(\lambda(p(s)))ds \\ & \text{s.t.} \quad \int_0^T \lambda(p(s))ds \le x \\ & \qquad p(s) \in [\underline{p},\overline{p}] \cup p_\infty \quad \forall s \in [0,T]. \end{aligned} \tag{3.4}$$

In (3.4) all the stochastic processes are substituted by their mean values. In [14], the authors showed that $J^D(x,T;\lambda)$ provides an upper bound on the expected revenue generated by any admissible pricing policy π, that is, $J^\pi(x,T;\lambda) \le J^D(x,T;\lambda)$, for all $\lambda \in \Gamma$ and $\pi \in \mathcal{P}$. Using this notion, we define the relative regret $R^\pi(x,T;\lambda)$ for any given demand function $\lambda \in \Gamma$ and policy $\pi \in \mathcal{P}$ to be

$$R^\pi(x,T;\lambda) = 1 - \frac{J^\pi(x,T;\lambda)}{J^D(x,T;\lambda)}. \tag{3.5}$$

As we mentioned above, the deterministic optimal solution $J^D(x,T;\lambda)$ provides an upper bound of the expected value of any policy π, therefore $R^\pi(x,T;\lambda)$ is between 0 and 1. And by definition, the smaller the regret, the closer π is to the optimal policy. However, since the decision maker does not know the true demand function, it is attractive to obtain a pricing policy π that achieves small regrets across all the underlying demand function $\lambda \in \Gamma$. In particular, we want to consider the "worst-case" regret, where the decision maker chooses a pricing policy π, and the nature picks the worst possible demand function for that policy:

$$\sup_{\lambda \in \Gamma} R^\pi(x,T;\lambda). \tag{3.6}$$

Obviously, the seller wants to minimize the worst-case regret, i.e., we are interested in solving:

$$\inf_{\pi \in \mathcal{P}} \sup_{\lambda \in \Gamma} R^\pi(x,T;\lambda). \tag{3.7}$$

Now our objective is clear. However, it is still hard to evaluate (3.7) for any single problem. Therefore, in this work, we adopt the widely-used asymptotic performance analysis (the regime of high volume of sales). We consider a regime in which both the size of the initial inventory, as well as the demand rate, grow proportionally large. In particular, for a market of size n, where n is a positive integer, the initial inventory

and the demand function are given by[1]:

$$x_n = nx \text{ and } \lambda_n(\cdot) = n\lambda(\cdot). \tag{3.8}$$

Denote $J_n^D(x, T; \lambda)$ to be the deterministic optimal solution for the problem with size n; it is easy to see that $J_n^D = nJ_1^D$. We also define $J_n^\pi(x, T; \lambda)$ to be the expected value of a pricing algorithm π when it is applied to a problem with size n. The relative regret for the size-n problem $R_n^\pi(x, T; \lambda)$ is:

$$R_n^\pi(x, T; \lambda) = 1 - \frac{J_n^\pi(x, T; \lambda)}{J_n^D(x, T; \lambda)}. \tag{3.9}$$

Our objective is to study the asymptotic behavior of $R_n^\pi(x, T; \lambda)$ as n grows large and design a mechanism that achieves small asymptotic regret.

3.4 Main Result: A Dynamic Learning Mechanism

In this section, we introduce our main result: a near-optimal dynamic learning mechanism. We first introduce some structural insights for this problem.

Consider the full-information deterministic problem (3.4). Since it is a deterministic optimization problem, it bears a deterministic optimal solution. As shown in [14], the optimal solution to (3.4) is given by

$$p(t) = p^D = \max\{p^u, p^c\} \tag{3.10}$$

where

$$p^u = arg \max_{p \in [\underline{p}, \overline{p}]} \{r(\lambda(p))\}, \tag{3.11}$$

$$p^c = arg \min_{p \in [\underline{p}, \overline{p}]} |\lambda(p) - \frac{x}{T}|. \tag{3.12}$$

[1]Here it is equivalent if we multiply n to both the inventory x and the time horizon T, the reason we choose x and $\lambda(\cdot)$ in this chapter is to make it consistent with previous literature.

Here, the superscript u stands for "unconstrained" and superscript c stands for "constrained". That is, the optimal solution to the deterministic problem (3.4) is to charge a constant price, which is the maximum between the price that maximizes the revenue rate and the price that exactly depletes the inventory, throughout the selling season. Moreover, as shown in [40], using this price p^D throughout the selling season will achieve a near-optimal regret even for the Poisson arrival model (with regret of order $O(n^{-1/2})$, which as will be shown later, is the regret we want to achieve). Therefore, the idea of our mechanism is to try to learn an estimate of p^D, using empirical observations at hand and hopefully without too much cost. We make one technical assumption about the value of p^D as follows.

Assumption B. There exists $\epsilon > 0$, such that $p^D \in [\underline{p} + \epsilon, \overline{p} - \epsilon]$ for all $\lambda \in \Gamma$.

Assumption B says that we require the optimal deterministic price to be in the interior of the initial price interval. This assumption is mainly for the purpose of analysis. One can always choose a large interval of $[\underline{p}, \overline{p}]$ to start with to guarantee that this assumption holds.

Now we state our main theorem:

Theorem 4 *Let Assumptions A and B hold for $\Gamma = \Gamma(M, K, m_L, m_U)$ and a fixed ϵ. Then there exists a policy π generated by Algorithm 3 (Dynamic Learning Mechanism), such that for all $n \geq 1$,*

$$\sup_{\lambda \in \Gamma} R_n^{\pi}(x, T; \lambda) \leq \frac{C(\log n)^{4.5}}{\sqrt{n}} \tag{3.13}$$

for some constant C depending on the coefficients in Γ, the initial inventory x, the length of time horizon T and ϵ.

A corollary of Theorem 4 follows from the relationship between the non-parametric model and the parametric one:

Corollary 1 *Assume* Γ *is a parameterized demand function family satisfying Assumption A and B for some coefficients. Then there exists a policy* π *generated by Algorithm 3, such that for all* $n \geq 1$*,*

$$\sup_{\lambda \in \Gamma} R_n^{\pi}(x, T; \lambda) \leq \frac{C(\log n)^{4.5}}{\sqrt{n}} \tag{3.14}$$

for some constant C*.*

We also establish a lower bound of the asymptotic regret for any admissible pricing policies:

Theorem 5 *There exists a set of demand functions* Γ *parameterized by a single parameter satisfying Assumption A and B with certain coefficients, such that for any admissible pricing policy* π*, for all* $n \geq 1$

$$\sup_{\lambda \in \Gamma} R_n^{\pi}(x, T; \lambda) \geq \frac{C}{\sqrt{n}} \tag{3.15}$$

for some constant C *that only depends on the coefficients in* Γ*,* x *and* T*.*

Remark 2 *Theorems 4 and 5 together provide a clear answer to the magnitude of regret the best pricing policy can achieve for this problem:* $O(n^{-1/2})$*. And this result holds for both parametric and non-parametric setting.*

Now we describe our mechanism. As we discussed in the beginning of this section, our mechanism aims to learn p^D. We achieve it through an iterative price experimentation. Specifically, our mechanism will be able to distinguish whether "p^u" or "p^c" is optimal, meanwhile it keeps shrinking a price interval containing p^D until a certain accuracy is achieved. We fist present the steps for our mechanism as below. Explanations and illustrations of the mechanism will follow right after.

Algorithm 3 (Dynamic Learning Mechanism) :

Step 1. Initialization:

(a) Consider a sequence of τ_i^u, κ_i^u, $i = 1, 2, ..., N^u$ and τ_i^c, κ_i^c, $i = 1, 2, ..., N^c$ (τ and κ represent the length of each learning period and the number of different prices tested in each learning period, respectively. Their values along with the value of N^u and N^c are defined in (3.29)-(3.33)). Define $\underline{p}_1^u = \underline{p}_1^c = \underline{p}$ and $\overline{p}_1^u = \overline{p}_1^c = \overline{p}$. Define $t_i^u = \sum_{j=1}^{i} \tau_j^u$, for $i = 0$ to N^u and $t_i^c = \sum_{j=1}^{i} \tau_j^c$, for $i = 0$ to N^c;

Step 2. Learn p^u or determine $p^c > p^u$:

For $i = 1$ to N^u do

(a) Divide $[\underline{p}_i^u, \overline{p}_i^u]$ into κ_i^u equally spaced intervals and let $\{p_{i,j}^u, j = 1, 2, ..., \kappa_i^u\}$ be the left endpoints of these intervals;

(b) Divide the time interval $[t_{i-1}^u, t_i^u]$ into κ_i^u equal parts and define

$$\Delta_i^u = \frac{\tau_i^u}{\kappa_i^u}, \qquad t_{i,j}^u = t_{i-1}^u + j\Delta_i^u, \qquad j = 0, 1, ..., \kappa_i^u;$$

(c) Apply $p_{i,j}^u$ from time $t_{i,j-1}^u$ to $t_{i,j}^u$, as long as the remaining inventory is still positive. If no more units are in stock, apply p_∞ until time T and STOP;

(d) Compute

$$\hat{d}(p_{i,j}^u) = \frac{\text{total demand over } [t_{i,j-1}^u, t_{i,j}^u)}{\Delta_i^u}, \quad j = 1, ..., \kappa_i^u;$$

(e) Compute

$$\hat{p}_i^u = arg \max_{1 \le j \le \kappa_i^u} \{p_{i,j}^u \hat{d}(p_{i,j}^u)\}$$

and

$$\hat{p}_i^c = arg \min_{1 \le j \le \kappa_i^u} |\hat{d}(p_{i,j}^u) - x/T|;$$

(f) If

$$\hat{p}_i^c > \hat{p}_i^u + 2\sqrt{\log n} \cdot \frac{\overline{p}_i^u - \underline{p}_i^u}{\kappa_i^u} \tag{3.16}$$

then BREAK from Step 2, Enter Step 3 and denote this i to be i_0;
Else, set $\hat{p}_i = \max\{\hat{p}_i^c, \hat{p}_i^u\}$. Define

$$\underline{p}_{i+1}^u = \hat{p}_i - \frac{\log n}{3} \cdot \frac{\overline{p}_i^u - \underline{p}_i^u}{\kappa_i^u} \tag{3.17}$$

and

$$\overline{p}_{i+1}^u = \hat{p}_i + \frac{2\log n}{3} \cdot \frac{\overline{p}_i^u - \underline{p}_i^u}{\kappa_i^u}. \tag{3.18}$$

And define the price range for the next iteration

$$I_{i+1}^u = [\underline{p}_{i+1}^u, \overline{p}_{i+1}^u].$$

Here we truncate the interval if it doesn't lie inside the feasible set $[\underline{p}, \overline{p}]$;

(g) If $i = N^u$, then Enter Step 4(a);

Step 3. Learn p^c when $p^c > p^u$:

For $i = 1$ to N^c do

(a) Divide $[\underline{p}_i^c, \overline{p}_i^c]$ into κ_i^c equally spaced intervals and let $\{p_{i,j}^c, j = 1, 2, ..., \kappa_i^c\}$ be the left endpoints of these intervals;

(b) Define

$$\Delta_i^c = \frac{\tau_i^c}{\kappa_i^c}, \qquad t_{i,j}^c = t_{i-1}^c + j\Delta_i^c + t_{i_0}, \qquad j = 0, 1, ..., \kappa_i^c;$$

(c) Apply $p_{i,j}^c$ from time $t_{i,j-1}^c$ to $t_{i,j}^c$, as long as the remaining inventory is still positive. If no more units are in stock, apply p_∞ until time T and STOP;

(d) Compute

$$\hat{d}(p_{i,j}^c) = \frac{\text{total demand over } [t_{i,j-1}^c, t_{i,j}^c)}{\Delta_i^c}, \quad j = 1, ..., \kappa_i^c;$$

(e) Compute

$$\hat{q}_i = arg \min_{1 \le j \le \kappa_i^c} |\hat{d}(p_{i,j}^c) - x/T|.$$

Define

$$\underline{p}^c_{i+1} = \hat{q}_i - \frac{\log n}{2} \cdot \frac{\overline{p}^c_i - \underline{p}^c_i}{\kappa^c_i} \tag{3.19}$$

and

$$\overline{p}^c_{i+1} = \hat{q}_i + \frac{\log n}{2} \cdot \frac{\overline{p}^c_i - \underline{p}^c_i}{\kappa^c_i}. \tag{3.20}$$

And define the price range for the next iteration

$$I^c_{i+1} = [\underline{p}^c_{i+1}, \overline{p}^c_{i+1}].$$

Here we truncate the interval if it doesn't lie inside the feasible set of $[\underline{p}, \overline{p}]$;

(f) If $i = N^c$, then enter Step 4(b);

Step 4. Apply the learned price:

(a) Define $\tilde{p} = \hat{p}_{N^u} + 2\sqrt{\log n} \cdot \frac{\overline{p}^u_{N^u} - \underline{p}^u_{N^u}}{\kappa^u_{N^u}}$. Use $\tilde{p}$ for the rest of the selling season until the stock runs out;

(b) Define $\tilde{q} = \hat{q}_{N^c}$. Use $\tilde{q}$ for the rest of the selling season until the stock runs out.

Now we explain this mechanism before we proceed to proofs. The idea of this mechanism is to divide the selling season into several periods and in each period, we test a grid of prices within a certain price interval. We find the empirical optimal price in each period, then shrink the price interval to a smaller one that still contains the optimal price (with high probability), and enter the next time period with the smaller price interval. We repeat the shrinking procedure until the price interval is small enough so that the desired accuracy is achieved.

Recall that the optimal deterministic price p^D is equal to the maximum of p^u and p^c, where p^u and p^c are solved from (3.11) and (3.12) respectively. It turns out that (3.11) and (3.12) have quite different local behaviors around its optimal solution under our assumptions: (3.11) resembles a quadratic function while (3.12) resembles a linear function. This difference requires us to have different shrinking strategies for the case when $p^u > p^c$ and $p^c > p^u$. This is why we have two learning steps (Step 2 and 3) in

our mechanism. Specifically, in Step 2, the mechanism works by shrinking the price interval until either a transition condition (3.16) is triggered or the learning phase is terminated. As will be shown later, when the transition condition (3.16) is triggered, with high probability, we are certain that the optimal solution to the deterministic problem is p^c. Otherwise, if we terminate learning before the condition is triggered, we know that p^u is either the optimal solution to the deterministic problem or it is close enough so that using p^u will also give us a near-optimal revenue. When the transition condition (3.16) happens, we switch to Step 3, where we use a new set of shrinking and price testing parameters. Note that in Step 3, we start from the initial price interval rather than the current interval obtained. This is solely for the ease of analysis, and in practice, one can simply continue from the last price interval. Both Step 2 and Step 3 must terminate in a finite number of iterations (we prove this in Lemma 11).

After this learning-while-doing period ends, a fixed price is used for the remaining selling season (Step 4) until the inventory runs out. To help illustration, a high-level description of the algorithm is shown below. One thing to note is that the "next" intervals defined in (3.17) and (3.18) are not symmetric. Similarly in Step 4(a), we use an adjusted price for the remaining selling season. This adjustment is a trick to make sure that the inventory consumption can be adequately upper bounded. Meanwhile the adjustment is small enough so that the revenue is maintained. The intuition is based on the different local behaviors of the revenue rate function and the demand rate function. The detailed reasoning of this adjustment will be clearly illustrated in Lemma 11.

High-level description of the Dynamic Learning Mechanism:

Step 1. Initialization:

(a) Initialize the time length and price granularity for each learning period. Set the maximum number of iterations N^u and N^c in Step 2 and 3;

Step 2. Learn p^u or determine $p^c > p^u$:

(a) Set the initial price interval to be $[\underline{p}, \overline{p}]$;

(b) Test a grid of prices on the current price interval for a predetermined length of time, observe the demand for each price;

(c) Compute the empirical optimal p^c and p^u using the observed demand;

(d) If p^c is "significantly" greater than p^u, then enter Step 3; otherwise, shrink the current interval to a subinterval containing the empirical optimal p^D;

(e) Repeat (b)-(d) for N^u times and then enter Step 4(a);

Step 3. Learn p^c when $p^c > p^u$:

(a) Set the initial price interval to be $[\underline{p}, \overline{p}]$;

(b) Test a grid of prices on the current price interval for a predetermined length of time, observe the demand for each price;

(c) Compute the empirical optimal p^c using the observed demand;

(d) Shrink the current interval to a subinterval containing the empirical optimal p^c;

(e) Repeat (b)-(d) for N^c times and then enter Step 4(b);

Step 4. Apply the learned price:

(a) Apply the last price in Step 2 until the stock runs out or the selling season finishes;

(b) Apply the last price in Step 3 until the stock runs out or the selling season finishes.

In the following, without loss of generality, we assume $T = 1$ and $\overline{p} - \underline{p} = 1$. Now we define $\tau_i^u, \kappa_i^u, N^u, \tau_i^c, \kappa_i^c$ and N^c. We first show a set of equations we want (τ_i^u, κ_i^u) and (τ_i^c, κ_i^c) to satisfy. Then we explain the meaning of each equation and solve those equations.

We want τ_i^u and κ_i^u to satisfy (here we use $f \sim g$ to mean that f and g are of the same order in n):

$$\left(\frac{\overline{p}_i^u - \underline{p}_i^u}{\kappa_i^u}\right)^2 \sim \sqrt{\frac{\kappa_i^u}{n\tau_i^u}} \quad ,\forall i = 1, ..., N^u, \tag{3.21}$$

$$\overline{p}_{i+1}^u - \underline{p}_{i+1}^u \sim \log n \cdot \frac{\overline{p}_i^u - \underline{p}_i^u}{\kappa_i^u} \quad ,\forall i = 1, ..., N^u - 1, \tag{3.22}$$

$$\tau_{i+1}^u \cdot \left(\frac{\overline{p}_i^u - \underline{p}_i^u}{\kappa_i^u}\right)^2 \cdot \sqrt{\log n} \sim \tau_1^u \quad ,\forall i = 1, ..., N^u - 1. \tag{3.23}$$

Also we define

$$N^u = \min_l \{l | \left(\frac{\overline{p}_l^u - \underline{p}_l^u}{\kappa_l^u}\right)^2 \sqrt{\log n} < \tau_1^u\}. \tag{3.24}$$

We then state the set of equations we want τ_i^c and κ_i^c to satisfy:

$$\frac{\overline{p}_i^c - \underline{p}_i^c}{\kappa_i^c} \sim \sqrt{\frac{\kappa_i^c}{n\tau_i^c}} \quad ,\forall i = 1, ..., N^c, \tag{3.25}$$

$$\overline{p}_{i+1}^c - \underline{p}_{i+1}^c \sim \log n \cdot \frac{\overline{p}_i^c - \underline{p}_i^c}{\kappa_i^c} \quad ,\forall i = 1, ..., N^c - 1, \tag{3.26}$$

$$\tau_{i+1}^c \cdot \frac{\overline{p}_i^c - \underline{p}_i^c}{\kappa_i^c} \cdot \sqrt{\log n} \sim \tau_1^c \quad ,\forall i = 1, ..., N^c - 1. \tag{3.27}$$

Also we define

$$N^c = \min_l \{l | \frac{\overline{p}_l^c - \underline{p}_l^c}{\kappa_l^c} \sqrt{\log n} < \tau_1^c\}. \tag{3.28}$$

Before we explain these desired relationships, let us first examine where the revenue loss comes from in this mechanism. First, in each period, there is an *exploration bias*, that is, the prices tested in each period may deviate from the optimal price, resulting in suboptimal revenue rate or suboptimal demand rate. These deviations multiplied by the time length of each period will be the "loss" for that period. Second, since

we only explore a grid of prices and may never encounter the exact optimal optimal price, therefore there is a *deterministic error* in the end. Third, since the demand is essentially a stochastic process, the observed demand rate may deviate from the true demand rate, resulting in a *stochastic error*. Note that these three errors also exist in the learning mechanism proposed in [14]. However, in our mechanism, each error does not simply appear once. For example, the deterministic error and stochastic error in one period may have impact on the errors in all the future periods. Thus, the design of our mechanism will revolve around the idea of balancing these errors in each step to achieve the maximum efficiency of learning. With this in mind, we explain the meaning of each equation above in the following:

- The first equation (3.21) ((3.25), resp.) balances the deterministic error induced by only considering the grid points (in which the grid granularity is $\frac{\overline{p}_i^u - \underline{p}_i^u}{\kappa_i^u}$ ($\frac{\overline{p}_i^c - \underline{p}_i^c}{\kappa_i^c}$, resp.)) and the stochastic error induced in the learning period which is $\sqrt{\frac{\kappa_i^u}{n\tau_i^u}}$ ($\sqrt{\frac{\kappa_i^c}{n\tau_i^c}}$, resp.). These two terms determine the price deviation in the next period and thus the exploration error of next period. We will show that under our assumptions, the loss due to price granularity is quadratic in Step 2, and linear in Step 3. We balance these two errors to achieve the maximum efficiency in learning.

- The second equation (3.22) ((3.26), resp.) is used to make sure that with high probability, our learning interval I_i^u (I_i^c, resp.) contains the optimal price p^D. We have to guarantee that I_i^u (I_i^c, resp.) contains p^D, otherwise we will incur a constant exploration error in all periods afterwards. This relationship is actually given in the definition of our mechanism (see (3.17), (3.18),(3.19) and (3.20)). However, we include them here for the sake of completeness.

- The third equation (3.23) ((3.27), resp.) is used to bound the exploration error for each learning period. This is done by considering the multiplication of the revenue rate deviation (also demand rate deviation) and the length of the learning period, which in our case is $\tau_{i+1}^u \sqrt{\log n} \cdot \left(\frac{\overline{p}_i^u - \underline{p}_i^u}{\kappa_i^u}\right)^2$ ($\tau_{i+1}^c \sqrt{\log n} \cdot \frac{\overline{p}_i^c - \underline{p}_i^c}{\kappa_i^c}$, resp.). We want this loss to be of the same order for each learning period (thus

all equal to the loss in the first learning period, which is τ_1) to achieve the maximum efficiency of learning.

- The fourth equation (3.24) ((3.28), resp.) determines if the price we obtain is guaranteed to be close enough to optimal such that we can apply this price in the remaining selling season. We show that $\sqrt{\log n} \cdot \left(\frac{\overline{p}_l^u - \underline{p}_l^u}{\kappa_l^u}\right)^2$ ($\sqrt{\log n} \cdot \frac{\overline{p}_l^c - \underline{p}_l^c}{\kappa_l^c}$, resp.) is the revenue rate and demand rate deviation of price $\hat{p}_l$. When this is less than τ_1, we can simply apply $\hat{p}_l$ and the loss will not exceed the loss of the first learning period.

Now we solve for κ_i^u and τ_i^u from the relations defined above.

Define $\tau_1^u = n^{-\frac{1}{2}} \cdot (\log n)^{3.5}$, one can solve (3.21), (3.22) and (3.23) and get:

$$\kappa_i^u = n^{\frac{1}{10}(\frac{3}{5})^{i-1}} \cdot \log n, \quad \forall i = 1, 2, ..., N^u, \tag{3.29}$$

$$\tau_i^u = n^{-\frac{1}{2}\cdot(\frac{3}{5})^{i-1}} \cdot (\log n)^5, \quad \forall i = 1, 2, ..., N^u. \tag{3.30}$$

And as a by-product, we have

$$\overline{p}_i^u - \underline{p}_i^u = n^{-\frac{1}{4}(1-(\frac{3}{5})^{i-1}))}, \quad \forall i = 1, 2, ..., N^u. \tag{3.31}$$

Next we do a similar computation for κ_i^c and τ_i^c. Define $\tau_1^c = n^{-\frac{1}{2}} \cdot (\log n)^{2.5}$. We have the following results:

$$\kappa_i^c = n^{\frac{1}{6}(\frac{2}{3})^{i-1}} \cdot \log n, \quad \forall i = 1, 2, ..., N^c, \tag{3.32}$$

$$\tau_i^c = n^{-\frac{1}{2}\cdot(\frac{2}{3})^{i-1}} \cdot (\log n)^3, \quad \forall i = 1, 2, ..., N^c, \tag{3.33}$$

and

$$\overline{p}_i^c - \underline{p}_i^c = n^{-\frac{1}{2}(1-(\frac{2}{3})^{i-1})}, \quad \forall i = 1, ..., N^c. \tag{3.34}$$

3.5 Proof Outlines

In this section, we give an outline of the proof of Theorem 4. We leave most of the detailed proof in Appendix B.3.

As the first step of our proof, we show that our mechanism will stop within a finite number of iterations. We have the following lemma:

Lemma 11 *N^u and N^c defined in (3.24) satisfy: $N^u \leq \log n$ and $N^c \leq \log n$.*

Proof. See Appendix B.3. □

Although Lemma 11 is simple, it is important since it provides an upper bound of the number of iterations of our mechanism. In our analysis, we frequently need to take a union bound over the number of iterations, and Lemma 11 will be used. In our mechanism, it is important to make sure that the deterministic optimal price p^D is always contained in our price interval. This is because when we miss the deterministic optimal price, we will incur a constant loss for all periods afterwards, and thus we will not not achieve the asymptotic optimality. The next lemma will show exactly such behavior of our mechanism.

Lemma 12 *Assume p^D is the optimal price for the deterministic problem and Assumptions A and B hold for $\Gamma = \Gamma(M, K, m_L, m_U)$ and $\epsilon > 0$. Then with probability $1 - O\left(\frac{1}{n}\right)$,*

- *If we never enter Step 3, then $p^D \in I_i^u$ for all $i = 1, 2, ..., N^u$;*
- *If Step 2 stops at i_0 and the algorithm enters Step 3, then $p^D \in I_i^u$ for all $i = 1, 2, ..., i_0$ and $p^D \in I_j^c$ for all $j = 1, 2, ..., N^c$.*

Proof. Here we give a sketch of the proof for the first part of this lemma. The detailed proof is given in Appendix B.3.

We prove by induction on i. Assume $p^D \in I_i^u$. We consider the $(i+1)$th iteration. Define

$$u_n^i = 2 \log n \cdot \max \left\{ \left(\frac{\overline{p}_i^u - \underline{p}_i^u}{\kappa_i^u} \right)^2, \sqrt{\frac{\kappa_i^u}{n\tau_i^u}} \right\}. \tag{3.35}$$

Denote the unconstrained and constrained optimal solutions on the current interval to be p_i^u and p_i^c. We can show (the details are in Appendix B.3) that with probability

$1 - O\left(\frac{1}{n^2}\right)$, $|\hat{p}_i^u - p_i^u| < C\sqrt{u_n^i}$ and $|\hat{p}_i^c - p_i^c| < C\sqrt{u_n^i}$ (in our analysis, for simplicity, we use C to denote a generic constant that only depends on Γ and ϵ). Therefore, with probability $1 - O\left(\frac{1}{n^2}\right)$, $|\hat{p}_i - p^D| < C\sqrt{u_n^i}$. On the other hand, the length of the next price interval (the center is near $\hat{p}_i$) is of order $\sqrt{\log n}$ greater than $\sqrt{u_i^n}$. Therefore, with probability $1 - O\left(\frac{1}{n^2}\right)$, $p^D \in I_{i+1}^u$. Then we take a union bound over all i (at most $\log n$) and the first part of the lemma holds. □

Next we show that if condition (3.16) is triggered, then with probability $1-O\left(\frac{1}{n}\right)$, $p^c > p^u$. An equivalent expression is that if $p^u \geq p^c$, then with probability $1-O\left(\frac{1}{n}\right)$, condition (3.16) will not be triggered.

Lemma 13 *If $p^u \geq p^c$, then with probability $1-O\left(\frac{1}{n}\right)$, our mechanism will not enter Step 3 before stopping.*

Proof. The proof of this lemma follows from the proof of Part 2 in Lemma 12. See Appendix B.3. □

Remark. Lemma 13 says that if $p^u \geq p^c$, then our mechanism will not enter Step 3. When $p^c > p^u$, however, it is also possible that our mechanism will not enter Step 3, but as we will show later, in that case, p^u must be very close to p^c so that the revenue collected is still near-optimal.

Now we have proved that with high probability, p^D will always be in our price interval. Next we analyze the revenue collected and prove our main theorem.

We first prove the case when $p^u \geq p^c$. We prove:

Proposition 3 *When $p^u \geq p^c$, $\sup_{\lambda\in\Gamma} R_n^\pi(x, T; \lambda) \leq \frac{C(\log n)^{4.5}}{\sqrt{n}}$.*

Consider a problem with size n. Recall that $J_n^\pi(x, T; \lambda)$ is the expected revenue collected by our mechanism given that the underlying demand function is λ. Define Y_{ij}^u to be Poisson random variables with parameters $\lambda(p_{i,j}^u)n\Delta_i^u$ ($Y_{ij}^u = N(\lambda(p_{i,j}^u)n\Delta_i^u))$[2].

[2]We remove the dependence on n in the notation. If not otherwise stated, it is assumed we are talking a problem with size n.

Also define $\hat{Y}^u$ to be a Poisson random variable with parameter $\lambda(\tilde{p})n(1-t^u_{N^u})$ ($\hat{Y}^u = N(\lambda(\tilde{p})n(1-t^u_{N^u}))$).

We define the following events ($I(\cdot)$ denotes the indicator function of a certain event):

$$A^u_1 = \{\omega : \sum_{i,j} Y^u_{ij} < nx\},$$

$$A^u_2 = \{\omega : \text{The algorithm never enters Step 3 and } p^D \in I^u_i, \forall i = 1, 2, ..., N^u\}.$$

We have

$$J^\pi_n(x, T; \lambda) \geq E[\sum_{i=1}^{N^u} \sum_{j=1}^{\kappa^u_i} p^u_{i,j} Y^u_{ij} I(A^u_1) I(A^u_2)] + E[\tilde{p} \min(\hat{Y}^u, (nx - \sum_{i,j} Y^u_{ij})^+) I(A^u_1) I(A^u_2)]. \tag{3.36}$$

In the following, we study each term in (3.36). We show that the revenue collected in both parts is "close" to the revenue generated by the optimal deterministic price p^D on that part (and the consumed inventory is also near-optimal). We first have:

Lemma 14

$$E[\sum_{i=1}^{N^u} \sum_{j=1}^{\kappa^u_i} p^u_{i,j} Y^u_{ij} I(A^u_1) I(A^u_2)] \geq \sum_{i=1}^{N^u} p^D \lambda(p^D) n \tau^u_i - Cn\tau^u_1 \log n. \tag{3.37}$$

Proof. This proof analyzes the exploration error in each iteration. Under A^u_2, the exploration error is quadratic in the length of the price interval of each period. Summing up those errors will lead to this result. The detail of the proof is given in Appendix B.3. □

Now we look at the other term in (3.36), we have

$$\begin{aligned} & E[\tilde{p} \min(\hat{Y}^u, (nx - \sum_{i,j} Y^u_{ij})^+) I(A^u_1) I(A^u_2)] \\ = \; & E[\tilde{p}(\hat{Y}^u - \max(\hat{Y}^u - (nx - \sum_{i,j} Y^u_{ij})^+, 0)) I(A^u_1) I(A^u_2)] \\ \geq \; & E[\tilde{p} \hat{Y}^u I(A^u_1) I(A^u_2)] - E[\tilde{p}(\hat{Y}^u + \sum_{i,j} Y^u_{ij} - nx)^+]. \end{aligned} \tag{3.38}$$

For the first term, we have

$$\begin{aligned} & E[\tilde{p}\hat{Y^u}I(A_1^u)I(A_2^u)] \\ = \; & E[n(1-t^u_{N^u})\tilde{p}\lambda(\tilde{p})I(A_1^u)I(A_2^u)] \\ \geq \; & (1-O\left(\frac{1}{n}\right))n(1-t^u_{N^u})E[\tilde{p}\lambda(\tilde{p})|I(A_1^u)I(A_2^u)]. \end{aligned} \tag{3.39}$$

However, by our assumption on the bound of the second derivative of $r(\lambda)$, and (3.23) and (3.24) we know that

$$E[\tilde{p}\lambda(\tilde{p})|I(A_1^u)I(A_2^u)] \geq p^D\lambda(p^D) - C(\overline{p}^u_{N^u+1} - \underline{p}^u_{N^u+1})^2 \geq p^D\lambda(p^D) - C\tau_1^u. \tag{3.40}$$

Therefore,

$$E[\tilde{p}\hat{Y^u}I(A_1^u)I(A_2^u)] \geq p^D\lambda(p^D)\cdot n(1-t^u_{N_u}) - Cn\tau_1^u. \tag{3.41}$$

Now we consider

$$E[\tilde{p}(\hat{Y}^u + \sum_{i,j} Y_{ij}^u - nx)^+]. \tag{3.42}$$

First we relax this to

$$\overline{p}E(\hat{Y}^u + \sum_{i,j} Y_{ij}^u - nx)^+. \tag{3.43}$$

We have the following lemmas:

Lemma 15

$$E(\hat{Y}^u + \sum_{i,j} Y_{ij}^u - E\hat{Y}^u - \sum_{i,j} EY_{ij}^u)^+ \leq Cn\tau_1^u \log n, \tag{3.44}$$

where C is a properly chosen constant.

Lemma 16 *If $p^u \geq p^c$, then*

$$\sum_{i=1}^{N^u}\sum_{j=1}^{\kappa_i^u} EY_{ij}^u + E\hat{Y}^u - nx \leq Cn\tau_1^u \log n, \tag{3.45}$$

where C is a properly chosen constant.

Proof of Lemma 15 and 16. The proof of Lemma 15 repeatedly uses Lemma 26 in Appendix B.2 which bounds the tail values of Poisson random variables. The proof of Lemma 16 bounds the inventory consumed in the learning period. We show that by the way we define each learning interval in (3.17) and (3.18), with high probability, p^c is always to the left of the center of the price interval. Therefore, the excess inventory we consumed in each iteration (compared to the consumption by p^c) is at most a second order quantity of the length of the price range (the first order error has been canceled out, or a negative value is remaining). This important fact will give us the desired bound for the consumption of the inventory. The detailed proofs of these two lemmas are given in Appendix B.3. □

Now combine Lemma 14, 15, and 16, we have

$$R_n^\pi \le 1 - \frac{np^D\lambda(p^D) - Cn\tau_1^u \log n}{np^D\lambda(p^D)} \le \frac{C(\log n)^{4.5}}{\sqrt{n}}. \tag{3.46}$$

Therefore, Proposition 3 follows. Next we consider the case when $p^c > p^u$. We claim

Proposition 4 *When $p^c > p^u$, $\sup_{\lambda\in\Gamma} R_n^\pi(x, T; \lambda) \le \frac{C(\log n)^{4.5}}{\sqrt{n}}$.*

When $p^c > p^u$, we condition our analysis on the time our mechanism enters Step 3. We define Y_{ij}^u and $\hat{Y}^u$ as before, and Y_{ij}^c to be Poisson random variables with parameters $\lambda(p_{i,j}^c)n\Delta_i^c$ ($Y_{ij}^c = N(\lambda(p_{i,j}^c)n\Delta_i^c)$). Also define $\hat{Y}_i^c$ to be a Poisson random variable with parameter $\lambda(\tilde{q})n(1 - t_{N^c}^c - t_i^u)$ ($\hat{Y}^c = N(\lambda(\tilde{q})n(1 - t_{N^c}^c - t_i^u))$).

We define the following events:

$$\begin{aligned} B_1 &= \{i_0 = 1\} \\ B_2 &= \{i_0 = 2\} \\ &\vdots \\ B_{N^u} &= \{i_0 = N^u\} \\ B_{N^u+1} &= \{\text{The algorithm doesn't enter Step 3}\}. \end{aligned} \tag{3.47}$$

Then we have the following bound on J_n^π in this case:

$$\begin{aligned} J_n^\pi(x,T,\lambda) \;\geq\; & E[\sum_{i=1}^{N^u}\sum_{j=1}^{\kappa_i^u} p_{i,j}^u Y_{ij}^u I(\cup_{l=i}^{N^u+1} B_l)] \\ & + E[\tilde{p}\min(\hat{Y}^u, (nx - \sum_{i,j} Y_{ij}^u)^+) I(B_{N^u+1})] + E[\sum_{i=1}^{N^c}\sum_{j=1}^{\kappa_i^c} p_{i,j}^c Y_{ij}^c I(\cup_{i=1}^{N^u} B_i)] \\ & + \sum_{l=1}^{N^u} E[\tilde{q}\min(\hat{Y}_l^c, (nx - \sum_{i=1}^{l}\sum_{j=1}^{\kappa_i^u} Y_{ij}^u - \sum_{i=1}^{N^c}\sum_{j=1}^{\kappa_i^c} Y_{ij}^c)^+) I(B^l)]. \end{aligned} \tag{3.48}$$

We will get a bound on each term. We prove the following lemmas:

Lemma 17

$$E[\sum_{i=1}^{N^u}\sum_{j=1}^{\kappa_i^u} p_{i,j}^u Y_{ij}^u I(\cup_{l=i}^{N^u+1} B_l)] \geq \sum_{i=1}^{N^u} n\tau_i p^D \lambda(p^D) P(\cup_{l=i}^{N^u+1} B_i) - Cn\tau_1^u \log n. \tag{3.49}$$

Lemma 18

$$E[\tilde{p}\min(\hat{Y}^u, (nx - \sum_{i,j} Y_{ij}^u)^+) I(B_{N^u+1})] \geq p^D \lambda(p^D) \cdot n(1 - t_{N^u}^u) P(B_{N^u+1}) - Cn\tau_1^u \log n. \tag{3.50}$$

Lemma 19

$$\sum_{i=1}^{N^c}\sum_{j=1}^{\kappa_i^c} E[p_{i,j}^c Y_{ij}^c I(\cup_{l=1}^{N^u} B_l)] \geq \sum_{i=1}^{N^c} n\tau_i^c p^D \lambda(p^D) P(\cup_{l=1}^{N^u} B_l) - Cn\tau_1^c \log n. \tag{3.51}$$

Lemma 20 *For each $l = 1, ..., N^u$,*

$$E[\tilde{q}\min(\hat{Y}_l^c, (nx - \sum_{i=1}^{l}\sum_{j=1}^{\kappa_i^u} Y_{ij}^u - \sum_{i=1}^{N^c}\sum_{j=1}^{\kappa_i^c} Y_{ij}^c)^+) I(B_l)] \geq np^D \lambda(p^D)(1 - t_l^u - t_{N^c}^c) P(B_l) - Cn\tau_1^u \log n. \tag{3.52}$$

Proof. The proofs of the above lemmas resemble the proofs for the case when $p^u \geq p^c$. They are given in Appendix B.3. □

We then combine Lemma 17, 18, 19 and 20, adding the right hand side together, and Proposition 4 follows. Theorem 4 thus follows from Proposition 3 and 4.

3.6 Lower Bound Example

In this section, we prove Theorem 5. We show that there exists a class of demand functions satisfying our assumptions, however no pricing policy can achieve an asymptotic regret less than $\Omega(\frac{C}{\sqrt{n}})$.

The proof involves statistical bounds on hypothesis testing, and it resembles the example discussed in [57] and [15]. The relationship and differences between our approach and theirs are discussed in the end of this section.

Proposition 5 *Define a set of demand functions as follows. Let* $\lambda(p;z) = 1/2+z-zp$ *where* z *is a parameter taking value in* $Z = [1/3, 2/3]$ *(we denote this demand function set by* Λ*). Assume that* $\underline{p} = 1/2$ *and* $\overline{p} = 3/2$*. Also assume that* $x = 2$ *and* $T = 1$*. Then we have*

- *This class of demand function satisfies Assumption A. Furthermore, for any* $z \in [1/3, 2/3]$*, the optimal price* p^D *always equals to* p^u *and* $p^D \in [7/8, 5/4]$*. Therefore, it also satisfies Assumption B with* $\epsilon = 1/4$*.*

- *For any admissible pricing policy* π*,*

$$\sup_{z\in Z} R_n^\pi(x, T; z) \geq \frac{1}{3(48)^2\sqrt{n}}, \quad \forall n. \tag{3.53}$$

First, we explain some intuitions behind this example. Note that all the demand functions in Λ cross at one common point, that is, when $p = 1$, $\lambda(p;z) = 1/2$. Such a price is called an "uninformative" price in [57]. When there exists an "uninformative" price, experimenting at that price will not gain information about the demand function. Therefore, in order to "learn" the demand function (i.e., the parameter z) and determine the optimal price, one must at least perform some price experiments at prices away from the uninformative price; on the other hand, when the optimal

price is indeed the uninformative price, doing price experimentations at a price away from the optimal price will incur some revenue losses. This tension is the key idea for this lower bound example. Before we proceed, we list some general properties of the demand function set Λ we defined in Proposition 5.

Lemma 21 *For the demand function defined in Proposition 5, denote the optimal price p^D under parameter z to be $p^D(z)$. We have:*

1. $p^D(z) = (1+2z)/(4z)$
2. $p^D(z_0) = 1$ *for* $z_0 = 1/2$
3. $\lambda(p^D(z_0); z) = 1/2$ *for all* z
4. $-4/3 \leq r''(p; z) \leq -2/3$ *for all* p, z
5. $|p^D(z) - p^D(z_0)| \geq \frac{|z-z_0|}{4}$
6. $p^D(z) = p^u(z)$ *for all* z.

Now in order to quantify the tension mentioned above, we need a notion of "uncertainty" about the unknown demand parameter z. For this, we use the K-L divergence over two probability measures for a stochastic process.

For any policy π, and parameter z, let $\mathcal{P}_z^\pi$ denote the probability measure associated with the observations (the process observed when using policy π) when the true demand function is $\lambda(p; z)$. We also denote the corresponding expectation operator by E_z^π.

Given z and z_0, the Kullback-Leibler (K-L) divergence between the two measures $P_{z_0}^\pi$ and P_z^π over time 0 to T is given by the following (we refer to [18] for this

definition):

$$\begin{aligned}
\mathcal{K}(\mathcal{P}^{\pi}_{z_0}, \mathcal{P}^{\pi}_{z}) &= E^{\pi}_{z_0}\left[\int_0^{T=1} n\lambda(p(s);z)\left[\frac{\lambda(p(s);z_0)}{\lambda(p(s);z)}\log\frac{\lambda(p(s);z_0)}{\lambda(p(s);z)} + 1 - \frac{\lambda(p(s);z_0)}{\lambda(p(s);z)}\right] ds\right] \\
&= E^{\pi}_{z_0}\left[\int_0^1 \Big\{ n(1/2+z_0-z_0p(s))\log\frac{1/2+z_0-z_0p(s)}{1/2+z-zp(s)} + \right. \\
&\quad \left. n(1/2+z-zp(s)) - n(1/2+z_0-z_0p(s))\Big\}\, ds\right] \\
&= E^{\pi}_{z_0}\left[\int_0^1 \Big\{ n(1/2+z_0-z_0p(s))\left(-\log\frac{1/2+z-zp(s)}{1/2+z_0-z_0p(s)} - 1\right) \right. \\
&\quad \left. + n(1/2+z-zp(s))\Big\}\, ds\right].
\end{aligned}$$

Note that the K-L divergence is a measure of distinguishability between probability measures: if two probability measures are close, then they have a small K-L divergence and vice versa. In terms of pricing policies, a pricing policy π is more likely to distinguish between the case when the parameter is z and the case when the parameter is z_0 if the quantity $\mathcal{K}(P^{\pi}_{z_0}, P^{\pi}_{z})$ is large.

Now we show the following lemma, which gives a lower bound of the regret induced by any policy in terms of the K-L divergence; this means a pricing policy that is better able to distinguish different parameters will also be more costly.

Lemma 22 *For any $z \in Z$, and any policy π setting price in $\mathcal{P}$,*

$$\mathcal{K}(\mathcal{P}^{\pi}_{z_0}, \mathcal{P}^{\pi}_{z}) \leq 24n(z_0 - z)^2 R^{\pi}_n(x, T; z_0), \tag{3.54}$$

where $z_0 = 1/2$ and $R^{\pi}_n(x, T; z_0)$ is the regret function defined in (3.5) with λ being $\lambda(p; z_0)$.

Proof. The proof attempts to bound the final term in (3.54) and is given in Appendix B.4. □

Now we have shown that in order to have a policy that is able to distinguish between two different parameters, one has to give up some portion of the revenue. In the following lemma, we show that on the other hand, if a policy is not able to

distinguish between two close parameters, then there will also be some significant losses:

Lemma 23 *Let π be any pricing policy that sets prices in $[\underline{p}, \overline{p}]$ and p_∞. Define $z_0 = 1/2$ and $z_1^n = z_0 + \frac{1}{4n^{1/4}}$ (note $z_1^n \in [1/3, 2/3]$ for all $n \geq 2$). We have for any $n \geq 2$*

$$R_n^\pi(x, T; z_0) + R_n^\pi(x, T; z_1^n) \geq \frac{1}{3(48)^2\sqrt{n}} e^{-\mathcal{K}(\mathcal{P}_{z_0}^\pi, \mathcal{P}_{z_1^n}^\pi)}. \tag{3.55}$$

Proof. The proof uses similar ideas as discussed in [15] and [57]. Here we give some sketches of the proof. We define two non-intersecting intervals around $p^D(z_0)$ and $p^D(z_1^n)$. We show that when the true parameter is z_0, pricing using p in the second interval will incur a certain loss and the same order of loss will be incurred if we use p in the first interval when the true parameter is z_1^n. At each time, we treat our policy π as a hypothesis test engine, that maps the historic data into two actions:

- Choose a price in the first interval
- Choose a price outside the first interval

Then we can represent the revenue loss during the selling season by the "accumulated probability" of committing errors in those hypothesis tests. However, by the theory of the hypothesis test, one can lower bound the probability of the errors for any decision rule. Thus we can obtain a lower bound of revenue loss for any pricing policy. The complete proof is referred to Appendix B.4. □

Now we combine Lemma 22 and 23. By picking z in Lemma 22 to be z_1^n and add (3.54) and (3.55) together, we have:

$$\begin{aligned} 2\{R_n^\pi(x, T; z_0) + R_n^\pi(x, T; z_1^n)\} &\geq \frac{3}{32\sqrt{n}} \mathcal{K}(\mathcal{P}_{z_0}^\pi, \mathcal{P}_{z_1^n}^\pi) + \frac{1}{3(48)^2\sqrt{n}} e^{-\mathcal{K}(\mathcal{P}_{z_0}^\pi, \mathcal{P}_{z_1^n}^\pi)} \\ &\geq \frac{1}{3(48)^2\sqrt{n}} (\mathcal{K}(\mathcal{P}_{z_0}^\pi, \mathcal{P}_{z_1^n}^\pi) + e^{-\mathcal{K}(\mathcal{P}_{z_0}^\pi, \mathcal{P}_{z_1^n}^\pi)}) \\ &\geq \frac{1}{3(48)^2\sqrt{n}}. \end{aligned}$$

The last inequality is because for any number $w > 0$, $w + e^{-w} \geq 1$. Therefore, we have shown that for any n, no matter what policy is used, we always have

$$\sup_{\lambda \in \Lambda} R_n^{\pi}(x, T; \lambda) \geq \frac{1}{3(48)^2\sqrt{n}},$$

and Proposition 5 is proved.

Remark. Our proof is similar to the proof of the corresponding worst case examples in [14] and [57], but different in several ways. First, in [14], they considered only finite possible prices (but possibly in a high-dimensional space). In our case, a continuous interval of prices is allowed. Therefore, the admissible policy in our case is much larger. And the K-L divergence function is thus slightly more sophisticated than the one used in their proof. In fact, the structure of our proof more closely resembles the one in [57] where they consider a worst-case example for a general parametric choice model. However, in their model, the time is discrete. Therefore, a discrete version of the K-L divergence is used and the analysis is based on the sum of the errors of different steps. In some sense, our analysis can be viewed as an extension of the proof in [57] into continuous case.

3.7 Numerical Results

In this section, we perform numerical tests of the dynamic learning mechanisms discussed in this chapter. Specifically, we compare the results by using our dynamic learning mechanism to the mechanism proposed in [14].

In our numerical tests, we consider two underlying demand functions. One is linear with $\lambda_1(p) = 30 - 3p$ and the other is exponential with $\lambda_2(p) = 80e^{-0.5p}$. These two demand functions are in accordance with the demand function chosen in the numerical tests in [14] where they considered $\tilde{\lambda}_1 = 30 - 3p$ and $\tilde{\lambda}_2 = 10e^{1-p}$. The reason that we change the coefficients in $\lambda_2(p)$ is that we want to examine two different cases for our algorithm, one with $p^c > p^u$ and the other with $p^u > p^c$. Note that with underlying demand function λ_1, we have $p^D = p^u = 5 > p^c = 3$, and with underlying

demand function λ_2, we have $p^D = p^c = 2\ln 4 > p^u = 2$. In both cases, we assume the initial inventory level is $x = 20$, the selling horizon $T = 1$, the initial price interval is $[\underline{p}, \overline{p}] = [0.1, 10]$ and test the performance of our mechanism for different n's ranging from 10^2 to 10^7. For each case, we run 10^3 independent simulations, comparing the average performance to the deterministic optimal solution (the standard deviation of the simulation result is quite insignificant thus only the mean value is concerned). We also make the following modifications to our mechanism in implementations:

- We remove the $\log n$ factor in τ_i^u and τ_i^c in our numerical study. Otherwise the factors $(\log n)^5$ and $(\log n)^3$ in (3.30) and (3.33) are too large for the cases we study. Since the $\log n$ factors are mainly used for analysis purposes, this modification is quite reasonable. In fact, this modification leads to better performance in revenues in the cases we study.

- Whenever our mechanism enters Step 3, instead of using $[\underline{p}, \overline{p}]$ as the initial interval as we stated in our mechanism, we use $[\underline{p}^u_{i_0}, \overline{p}^u_{i_0}]$, which is the last computed price interval. As we showed in Lemma 12, with high probability, this interval contains the deterministic optimal price, therefore it intuitively shares the same theoretic performance guarantee. This would also make improvements to the performance of our mechanism.

Before we show our comparison results, we show a sample run of our mechanism on one of the tested problem to illustrate how it works and summarize some qualitative features of our mechanism. We consider the case with linear demand function $\lambda_1(p) = 30 - 3p$ and $n = 10^5$. A sketch of a single run of our mechanism is shown in Figure 3.1.

In Figure 3.1, we can see that our mechanism runs 4 iterations of price learning before entering the last step. The time spent in each iteration is increasing, which is in accordance with our definition of τ_i^u and is also intuitively true: since we are using a more accurate price for experimentation in each iteration, we can afford to spend more time without incurring extra losses. Besides, the number of prices tested in each interval is decreasing, along with the length of the price interval and the

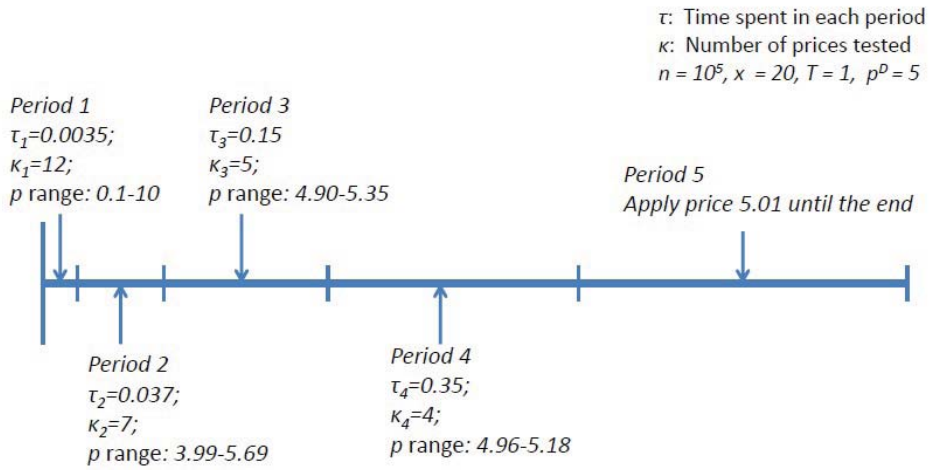
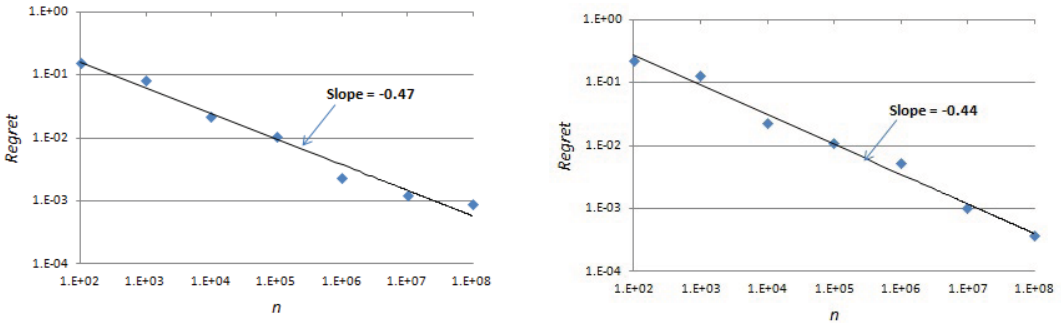


Figure 3.1: A sample run of the dynamic learning mechanism

price granularity. Therefore, with time evolving, we are testing fewer prices on each interval, on finer grids, and test longer for each price. And finally, we apply the last learned price in the rest of the time horizon.

Remember that we evaluate our mechanism by the regret function $R_n^\pi(x, T; \lambda)$ defined in (3.9). In Theorem 4, we showed that asymptotically, the regret is of order $n^{-1/2}$ (with a logarithmic factor). In other words, $\log(R_n^\pi(x, T; \lambda))$ should be approximately a linear function of $\log n$ with slope -0.5. In the following, we conduct numerical experiments for problems with different sizes of n and study how R_n^π changes with n. Specifically, we use a linear regression to fit the relationship between $\log R_n^\pi(x, T; \lambda)$ and $\log n$. The results are shown in Figure 3.2(a) and 3.2(b).

In Figure 3.2(a) and 3.2(b), the slopes of the best linear fit of the log-regret and $\log n$ are approximately 0.444 and 0.465, respectively. Although it is somewhat less than 0.5 as stated in our theorem, it is significantly larger than the slope of 0.25 obtained in [14] for the non-parametric policy, and even the slope of 0.33 obtained for the parametric policy in [14]. The deviation from 0.5, however, may be due to the ignorance of the log-factor which is not insignificant in the cases we study. We show the comparison of the performance between our dynamic learning mechanism



(a) Regret vs n with $\lambda_1 = 30 - 3p$

(b) Regret vs n with $\lambda_2 = 80e^{-0.5p}$

Figure 3.2: Asymptotic performance of the dynamic learning mechanism

and the one-time learning mechanism in [14] in Figure 3.3.

In Figure 3.3, we can see that under the non-parametric assumption, the dynamic learning mechanism does not only have a better asymptotic performance than the one-time learning mechanism, but also perform better in absolute regrets when n is greater than 100. This verifies the improvement of dynamic learning mechanism over the one-time learning mechanism.

Lastly we compare the non-parametric learning policy to a set of parametric learning policies. We consider the underlying demand function $\lambda(p) = 80e^{-0.5p}$ and all other settings remain the same as in previous tests. We first compare the performance of our dynamic learning policy to the parametric learning policy shown in [14]. Then we consider the case when the model is misspecified, that is, a linear demand function is assumed and parametric learning is performed under such assumption (using one-time learning in [14]). The results are summarized in Figure 3.4.

In Figure 3.4, we can see that the parametric approach in [14] performs better for small-sized problems (when model is well specified). However, when n goes large, our mechanism catches up and eventually surpasses the one-time learning parametric approach since we have a superior asymptotic performance. Also, we can see that the risk of model misspecification could indeed be quite severe. In Figure 3.4, the relative regret of the mechanism when the model is misspecified does not even converge to

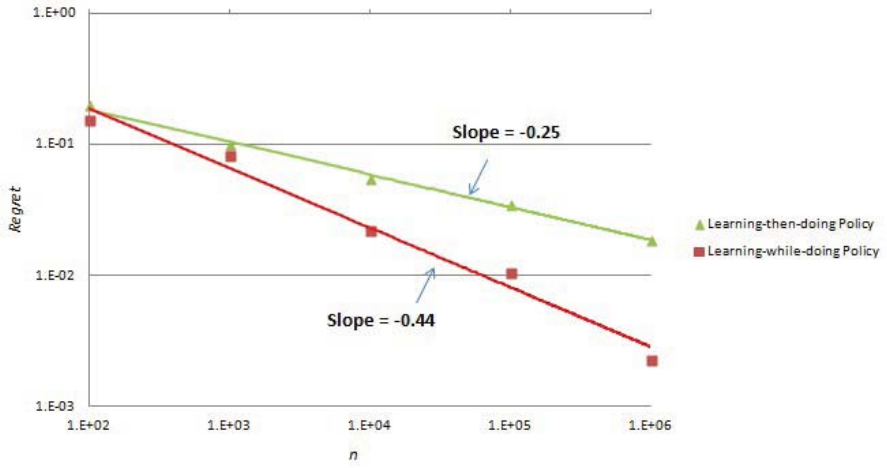


Figure 3.3: Performance comparison between dynamic learning and one-time learning mechanisms for non-parametric model

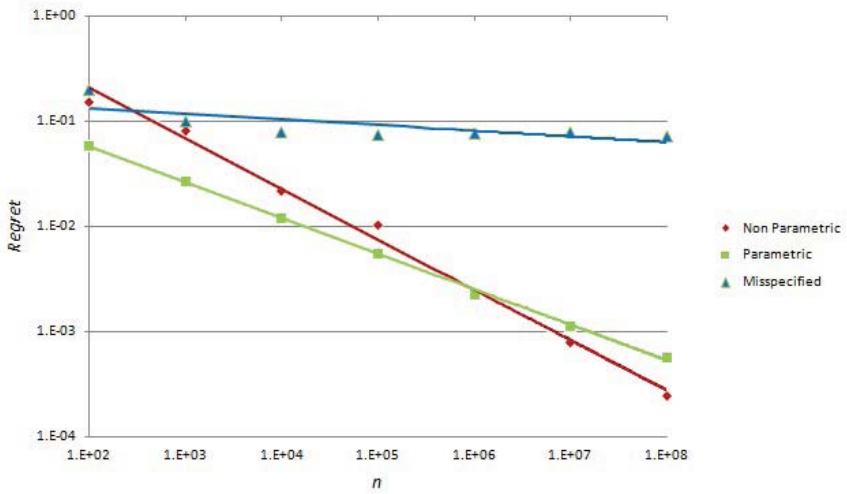


Figure 3.4: Performance comparison between non-parametric and parametric mechanisms and the risk of model misspecification

zero as n goes large. Similar phenomena are also shown in the numerical results in [14] and [16].

3.8 Extensions

Other Applications

In addition to the pricing problem we discussed in this chapter, our work can be applied to several other problems. In this section, we shed some light on the potential applications. We take a case where the firm chooses the advertisement intensity as example.

Consider a firm selling a single product over a finite time horizon. Due to the market competition, the price of the product has to be fixed. However, the company can choose its advertisement strategy α to affect the demand rate. For example, in the online selling case, α may be the pay-per-click price the company pays to the search engines. The demand rate under advertisement intensity α is denoted by $\lambda(\alpha)$. In this setting, the company controls α_t and the net revenue collected is:

$$\int_{t=0}^{T} (p - \alpha_t) dN^t, \tag{3.56}$$

where $N^t = N(\int_{s=0}^{t} \lambda(\alpha_t) dt)$ is a Poisson random variable. Consider the following transformation:

$$w_t = p - \alpha_t,$$

$$\tilde{\lambda}(w_t) = \lambda(\alpha_t).$$

Then this problem will have the same form of the optimal pricing problem we discussed in this chapter. Therefore, when the demand function satisfies the same set of conditions, our theorem will apply.

Besides advertisement intensity, the control variable can be viewed as the sales person compensation or other incentives of selling a product, as long as a similar formulation can be established. We believe there are more applications that fit into this model.

When the Second-order Derivative Assumption is not Satisfied

In Assumption A, we assumed that $r''(\lambda)$ exists and is bounded away from zero. This assumption is necessary in our analysis (at least locally at p^u) since we utilize the local quadratic behavior of the revenue functions. However, this assumption does not always hold, e.g., when the demand function is piecewise linear and the revenue maximizing price p^u is exactly at one of the "kink" points of the piecewise function. In that case, at $\lambda(p^u)$, $r(\lambda)$ behaves more like a linear function. A natural question is: can we still achieve the same asymptotic behavior for those cases? The following theorem gives an assertive answer to this question, although it requires us to use another mechanism (Algorithm 4, see Appendix B.5) to achieve this.

Theorem 6 *Let Assumption A hold except for the third requirement. Let Assumption B hold for a fixed $\epsilon > 0$. Also, assume that for any $p \in [\underline{p}, \overline{p}]$, $L|p - p^u| \leq |r(\lambda(p)) - r(\lambda(p^u))|$. Then there exists a policy $\pi \in \mathcal{P}$ generated by Algorithm 4, such that for all $n \geq 1$,*

$$\sup_{\lambda \in \Gamma} R_n^\pi(x, T; \lambda) \leq \frac{C(\log n)^3}{\sqrt{n}}, \tag{3.57}$$

for some constant C.

Theorem 6 complements our main theorem for the cases when the demand function is not differentiable. In fact, in this case, a simpler learning algorithm (see Algorithm 4) involving only one learning step would work. However, to apply this theorem requires the knowledge that the demand function has a "kink" at optimal. A unified mechanism that works for both cases is still one of our future research work.

3.9 Conclusions

In this chapter, we present a dynamic learning mechanism for a class of single-product revenue management problems. Our mechanism achieves an asymptotic regret of $O(n^{-1/2})$ (with some logarithmic factor) even if we have no prior knowledge on the demand function except for some regularity conditions. By complementing with a worst-case bound, we show that our mechanism is almost the best possible in this

setting, and it closes the performance gaps between parametric and non-parametric learning and between a posted-price model and a customer-bidding model.

In this chapter, we show that learning and doing need not to be separated. In fact, it should not be separated as an integrated learning and doing procedure is proved to achieve near-best performance. This design of learning mechanisms may be of independent interest to the revenue management practitioners.

There are several open questions to explore. One of them is how to extend this result to high-dimensional problems. In high-dimensional problems, the structure of demand functions may be even more complicated and a price experimentation idea like the one used in this chapter may not work due to the curse of dimensionality. Therefore, the extension is not straightforward. Other directions may include models with competition among retailers and/or strategic behaviors of the customers.

Chapter 4

Conclusions and Future Work

In this dissertation, we proposed a class of dynamic learning mechanisms and applied them to two revenue management models: one customer bidding model and one posted-price model. We showed that our mechanisms is robust, dynamic and optimal, and all of which can be attributed to one important feature of our mechanism: dynamic learning. This feature enables our mechanism to outperform one-time learning mechanisms and may have important implications in practice.

There are several directions for future research. From technical point of view, there are still small gaps between the proved performance of our mechanisms and the worst-case bound. As suggested by our numerical results, this gap may be closed by a more careful analysis of our mechanism. Also, it is an interesting question of how to extend the learning mechanism into more general cases, e.g., high-dimensional cases. From practical point of view, how to integrate dynamic learning with other factors of revenue management problems such as customer choice behavior and competition are all important directions for future research.

Appendix A

Supporting Proofs for Chapter 2

A.1 Supporting Proofs for Section 2.4

Hoeffding-Bernstein's Inequality

Lemma 24 [Hoeffding-Bernstein's Inequality, Theorem 2.14.19 in [64]] *Let* $u_1, u_2, ...u_r$ *be random samples without replacement from the real numbers* $\{c_1, c_2, ..., c_R\}$. *Then for every* $t > 0$,

$$P(|\sum_i u_i - r\bar{c}| \geq t) \leq 2\exp(-\frac{t^2}{2r\sigma_R^2 + t\Delta_R}), \tag{A.1}$$

where $\Delta_R = \max_i c_i - \min_i c_i$, $\bar{c} = \frac{1}{R}\sum_i c_i$, *and* $\sigma_R^2 = \frac{1}{R}\sum_{i=1}^{R}(c_i - \bar{c})^2$.

Proof of Inequality (2.13)

We prove that with probability $1 - \epsilon$,

$$\hat{b}_i = \sum_{t \in N} a_{it} x_t(\hat{\boldsymbol{p}}) \geq (1 - 3\epsilon) b_i,$$

given $\sum_{t \in S} a_{it} x_t(\hat{\boldsymbol{p}}) \geq (1 - 2\epsilon)\epsilon b_i$. The proof is very similar to the proof of Lemma 2. Fix a price vector $\boldsymbol{p}$ and i. Define a permutation is "bad" for $\boldsymbol{p}, i$ if both (a) $\sum_{t \in S} a_{it} x_t(\boldsymbol{p}) \geq (1 - 2\epsilon)\epsilon b_i$ and (b) $\sum_{t \in N} a_{it} x_t(\boldsymbol{p}) \leq (1 - 3\epsilon) b_i$ hold.

Define $Y_t = a_{it}x_t(\boldsymbol{p})$. Then, the probability of bad permutations is bounded by:

$$P\left(|\sum_{t\in S} Y_t - \epsilon\sum_{t\in N} Y_t| \geq \epsilon^2 b_i | \sum_{t\in N} Y_t \leq (1-3\epsilon)b_i\right) \leq 2\exp\left(-\frac{b_i\epsilon^3}{3}\right) \leq \frac{\epsilon}{m\cdot n^m} \quad \text{(A.2)}$$

where the last inequality follows from the condition that $b_i \geq \frac{6m\log(n/\epsilon)}{\epsilon^3}$. Summing over n^m "distinct" prices and $i = 1, \ldots, m$, we get the desired inequality.

A.2 Supporting Proofs for Section 2.5

Proof of Lemma 5

Consider the i^{th} component of $\sum_t a_{it}\hat{x}_t$ for a fixed i. For ease of notation, we temporarily omit the subscript i. Define $Y_t = a_t x_t(\boldsymbol{p})$. If $\boldsymbol{p}$ is an optimal dual solution for (2.15), then by the complementarity conditions, we have:

$$\sum_{t=1}^{\ell} Y_t = \sum_{t=1}^{\ell} a_t x_t(\boldsymbol{p}) \leq (1-h_\ell)b\frac{\ell}{n}. \quad \text{(A.3)}$$

Therefore, the probability of "bad" permutations is bounded by:

$$\begin{aligned} & P(\sum_{t=1}^{\ell} Y_t \leq (1-h_\ell)\frac{b\ell}{n}, \sum_{t=\ell+1}^{2\ell} Y_t \geq \frac{b\ell}{n}) \\ \leq\ & P(\sum_{t=1}^{\ell} Y_t \leq (1-h_\ell)\frac{b\ell}{n}, \sum_{t=1}^{2\ell} Y_t \geq \frac{2b\ell}{n}) + P(|\sum_{t=1}^{\ell} Y_t - \frac{1}{2}\sum_{t=1}^{2\ell} Y_t| \geq \frac{h_\ell}{2}\frac{b\ell}{n}, \sum_{t=1}^{2\ell} Y_t \leq \frac{2b\ell}{n}). \end{aligned}$$

Define $\delta = \frac{\epsilon}{m \cdot n^m \cdot E}$. Using Hoeffding-Bernstein's Inequality (Lemma 24, here $R = 2\ell$, $\sigma_R^2 \leq b/n$, and $\Delta_R \leq 1$), we have:

$$\begin{aligned} P(\sum_{t=1}^{\ell} Y_t \leq (1-h_\ell)\frac{b\ell}{n}, \sum_{t=1}^{2\ell} Y_t \geq \frac{2b\ell}{n}) &\leq P(\sum_{t=1}^{\ell} Y_t \leq (1-h_\ell)\frac{b\ell}{n} | \sum_{t=1}^{2\ell} Y_t \geq \frac{2b\ell}{n}) \\ &\leq P(\sum_{t=1}^{\ell} Y_t \leq (1-h_\ell)\frac{b\ell}{n} | \sum_{t=1}^{2\ell} Y_t = \frac{2b\ell}{n}) \\ &\leq P(|\sum_{t=1}^{\ell} Y_t - \frac{1}{2}\sum_{t=1}^{2\ell} Y_t| \geq h_\ell \frac{b\ell}{n} | \sum_{t=1}^{2\ell} Y_t = \frac{2b\ell}{n}) \\ &\leq 2\exp(-\frac{\epsilon^2 b}{2+h_l}) \leq \frac{\delta}{2} \end{aligned}$$

and

$$P(|\sum_{t=1}^{\ell} Y_t - \frac{1}{2}\sum_{t=1}^{2\ell} Y_t| \geq \frac{h_\ell}{2}\frac{b\ell}{n}, \sum_{t=1}^{2\ell} Y_t \leq \frac{2b\ell}{n}) \leq 2\exp(-\frac{\epsilon^2 b}{8+2h_\ell}) \leq \frac{\delta}{2},$$

where the last steps hold because $h_\ell \leq 1$, and the condition made on B.

Next, we take a union bound over n^m distinct prices, $i = 1, \ldots, m$, and E values of ℓ, the lemma is proved.

Proof of Inequality (2.19)

The proof is very similar to the proof of Lemma 5. Fix a $\boldsymbol{p}$, ℓ and $i \in \{1, \ldots, m\}$. Define "bad" permutations for $\boldsymbol{p}, i, \ell$ as those permutations such that all the following conditions hold: (a) $\boldsymbol{p} = \hat{\boldsymbol{p}}^\ell$, that is, $\boldsymbol{p}$ is the price learned as the optimal dual solution for (2.15), (b) $p_i > 0$, and (c) $\sum_{t=1}^{2\ell} a_{it} x_t(\boldsymbol{p}) \leq (1 - 2h_\ell - \epsilon)\frac{2\ell}{n} b_i$. We show that the probability of these bad permutations is small.

Define $Y_t = a_{it} x_t(\boldsymbol{p})$. If $\boldsymbol{p}$ is an optimal dual solution for (2.15), and $p_i > 0$, then by the KKT conditions the i^{th} inequality constraint holds with equality. Therefore, by observation made in Lemma 1, we have:

$$\sum_{t=1}^{\ell} Y_t = \sum_{t=1}^{\ell} a_{it} x_t(\boldsymbol{p}) \geq (1-h_\ell)\frac{\ell}{n} b_i - m \geq (1 - h_\ell - \epsilon)\frac{\ell}{n} b_i, \tag{A.4}$$

where the second last inequality follows from $B = \min_i b_i \geq \frac{m}{\epsilon^2}$, and $\ell \geq \epsilon n$. Therefore, the probability of "bad" permutations for $\boldsymbol{p}, i, \ell$ is bounded by:

$$P\left(|\sum_{t=1}^{\ell} Y_t - \frac{1}{2}\sum_{t=1}^{2\ell} Y_t| \geq h_\ell \frac{b_i \ell}{n} | \sum_{t=1}^{2\ell} Y_t \leq (1 - 2h_\ell - \epsilon)\frac{2\ell}{n} b_i\right) \leq 2\exp\left(-\frac{\epsilon^2 b_i}{2}\right) \leq \delta$$

where $\delta = \frac{\epsilon}{m \cdot n^m \cdot E}$. The last inequality follows from the condition on B. Next, we take a union bound over the n^m "distinct" $\boldsymbol{p}$'s, $i = 1, \ldots, m$, and E values of ℓ, we conclude that with probability $1 - \epsilon$,

$$\sum_{t=1}^{2\ell} a_{it}\hat{x}_t(\hat{\boldsymbol{p}}^\ell) \geq (1 - 2h_\ell - \epsilon)\frac{2\ell}{n} b_i$$

for all i such that $\hat{\boldsymbol{p}}_i > 0$ and all ℓ.

A.3 Detailed Steps for Theorem 2

Let $c_1, \ldots, c_n$ denote the n customers. For each i, the set $R_i \subseteq \{c_1, \ldots, c_n\}$ of customers with bid vector $\boldsymbol{w}_i$ and bid 1 or 3 is fixed. $|R_i| = 2B/z$ for all i. Conditional on set R_i the bid values of customers $\{c_j, j \in R_i\}$ are independent random variables that take value 1 or 3 with equal probability.

Now consider t^{th} bid $(2, \boldsymbol{w_i})$. In at least 1/2 of the random permutations, the number of bids from set R_i before the bid t is less than B/z. Conditional on this event, with a constant probability the bids in R_i before t take values such that the bids after t can make the number of $(3, \boldsymbol{w}_i)$ bids more than B/z with a constant probability and less than $B/z - \sqrt{B/4z}$ with a constant probability. This probability calculation is similar to the one used by Kleinberg [45] in his proof of necessity of condition $B \geq \Omega(1/\epsilon^2)$. For completeness, we derive it in the Lemma 25 towards the end of the proof.

Now, in the first kind of instances (where number of $(3, \boldsymbol{w}_i)$ bids are more than B/z) retaining $(2, \boldsymbol{w}_i)$ is a "potential mistake" of 1, and in second kind, skipping $(2, \boldsymbol{w}_i)$ is a potential mistake of 1. We call it a potential mistake because it will cost

a revenue loss of 1 if the online algorithm decides to pick B/z of $\boldsymbol{w}_i$ bids. $|r_i - B/z|$ of these mistakes may be recovered in each instance by deciding to pick $r_i \neq B/z$ of $\boldsymbol{w}_i$ bids. The total expected number of potential mistakes is $\Omega(\sqrt{Bz})$ (since there are $\sqrt{B/4z}$ of $(2, \boldsymbol{w}_i)$ bids for every i). By Claim 1, no more than a constant fraction of instances can recover more than $7\epsilon B$ of the potential mistakes. Let ONLINE denote the expected value for the online algorithm over random permutation and random instances of the problem. Therefore,

$$\text{ONLINE} \leq \text{OPT} - \Omega(\sqrt{zB} - 7\epsilon B).$$

Now, observe that $\text{OPT} \leq 7B$. This is because by construction every set of demand vectors (consisting of either $\boldsymbol{v}_i$ or $\boldsymbol{w}_i$ for each i) will have at least 1 item in common, and since there are only B units of this item available, at most $2B$ demand vectors can be accepted giving a profit of at most $7B$. Therefore, $\text{ONLINE} \leq \text{OPT}(1 - \Omega(\sqrt{z/B} - 7\epsilon))$, and in order to get $(1-\epsilon)$ approximation factor we need

$$\Omega(\sqrt{z/B} - 7\epsilon) \leq O(\epsilon) \Rightarrow B \geq \Omega(z/\epsilon^2).$$

This completes the proof of Theorem 2.

Lemma 25 *Consider $2k$ random variables $Y_j, j = 1, \ldots, 2k$ that take value $0/1$ independently with equal probability. Let $r \leq k$. Then with constant probability $Y_1, \ldots, Y_r$ take value such that $\sum_{j=1}^{2k} Y_j$ can be greater or less than its expected value k by $\sqrt{k}/2$ with equal constant probability.*

$$Pr(|\sum_{j=1}^{2k} Y_j - k| \geq \lceil \sqrt{k}/2 \rceil \mid Y_1, \ldots Y_r) \geq c$$

for some constant $0 < c < 1$.

Proof.

- Given $r \leq k$, $|\sum_{j \leq r} Y_j - r/2| \leq \sqrt{k}/4$ with constant probability (by central limit theorem).

- Given $r \leq k$, $|\sum_{j>r} Y_j - (2k-r)/2)| \geq 3\sqrt{k}/4$ with constant probability.

Given the above events $|\sum_j Y_j - k| \geq \sqrt{k}/2$, and by symmetry both events have equal probability. □

A.4 Competitive Ratio in terms of OPT

In this section, we prove Proposition 2 that provides a lower bound condition depending only on OPT (and not on B). We also show that the conditions on OPT and B don't imply each other by giving examples showing that either of the conditions could be stronger depending on the problem instance.

Proof of Proposition 2: We only give a brief proof since the main idea is very similar to our main theorem but only with some adjustment of the parameters. Without loss of generality, we assume $\pi_{\max} = 1$. We still take $L = \{\epsilon n, 2\epsilon n, ...\}$, and compute the dual price of the following linear program for each $\ell \in L$, which we apply on the inputs from $\ell + 1$ to 2ℓ:

$$\begin{array}{lll} \text{maximize} & \sum_{t=1}^{\ell} \pi_t x_t & \\ \text{subject to} & \sum_{t=1}^{\ell} a_{it} x_t \leq (1 - h_{\ell,i})\frac{\ell}{n} b_i, & i = 1, \ldots, m \\ & 0 \leq x_t \leq 1, & t = 1, \ldots, \ell. \end{array} \tag{A.5}$$

Here, instead of using $h_\ell = \epsilon\sqrt{\frac{n}{\ell}}$ as in (2.15), we are using $h_{\ell,i} = \sqrt{\frac{2nm\log(n/\epsilon)}{\ell b_i}}$. Then by the same argument as we used in the proof of our main theorem, with probability at least $1 - O(\epsilon)$, none of the constraints will be violated and $\sum_{t=1}^{2\ell} a_{it} x_t(\hat{\boldsymbol{p}}^\ell) \geq (1 - 2h_{\ell,i} - \epsilon)\frac{2\ell}{n} b_i$ for all ℓ and i. Therefore, comparing to the optimal value of

$$\begin{array}{lll} \text{maximize} & \sum_{t=1}^{2\ell} \pi_t x_t & \\ \text{subject to} & \sum_{t=1}^{2\ell} a_{it} x_t \leq (1 - 2h_\ell - \epsilon)\frac{2\ell}{n} b_i & i = 1, ..., m \\ & 0 \leq x_t \leq 1 & t = 1, ..., 2\ell, \end{array} \tag{A.6}$$

the difference in value of new algorithm is at most $\sum_i (2h_{\ell,i} - 2h_\ell)^+ \frac{2l}{n} b_i$ (we have

assumed that $\pi_{max} = 1$). Also, we know by the proof of Theorem 1, using (A.6) gives a value of at least $(1 - O(\epsilon))$OPT, therefore the loss of the new algorithm is at most

$$O(\epsilon OPT) + 2\sum_{\ell,i}(2h_{\ell,i} - 2h_\ell)^+ \frac{\ell}{n} b_i \leq O(\epsilon OPT) + O\left(\frac{m^2 \log(n/\epsilon)}{\epsilon}\right) = O(\epsilon OPT).$$

The last inequality is because

$$\begin{aligned}
\sum_{\ell,i}(h_{\ell,i} - h_\ell)^+ \frac{\ell}{n} b_i &\leq m \sum_{\ell} \sup_b \left(\sqrt{\frac{2m\ell \log(n/\epsilon) b}{n}} - b\epsilon\sqrt{\frac{\ell}{n}}\right) \\
&\leq \sum_{\ell} \sqrt{\frac{\ell}{n}} \frac{m^2 \log(n/\epsilon)}{\epsilon} \\
&\leq O\left(\frac{m^2 \log(n/\epsilon)}{\epsilon}\right). \quad \square
\end{aligned}$$

Next, we compare Theorem 1 and Proposition 2 by providing examples where conditions for one is satisfied but not the other.

Example 1 Condition on B fails but condition on $\frac{OPT}{\pi_{max}}$ holds.
We start from any instance for which both conditions hold. However, we add another item to this instance, which has very few copies and contributes very little to the optimal solution (much less than OPT/m). Then condition on B will fail but the $\frac{OPT}{\pi_{max}}$ one still holds to guarantee that our algorithm returns a near-optimal solution. Intuitively, condition on $\frac{OPT}{\pi_{max}}$ works better when the coefficients π_is are uniform but the number of items b_i is highly non-uniform.

Example 2 Condition on $\frac{OPT}{\pi_{max}}$ fails but condition on B holds.
Again, we start from any instance for which both the conditions hold. Now we replace one coefficient π_i to take a value that equals to the original optimal objective value. Now, $\frac{OPT}{\pi_{max}}$ is less than 2, but the value of B stays unchanged. Intuitively, condition on B condition works better when the number of items are close to uniform but the coefficients π_is are not.

The above examples show that the conditions on B and OPT don't imply each

other. However, since the condition on B is checkable, while the condition on OPT is not, the former one might be preferable in practice.

A.5 General Constraint Matrix Case

Proof of Lemma 8

The proof is very similar to the proof of Lemma 5. To start with, we fix $\boldsymbol{p}$ and ℓ. This time, we say a random order is "bad" for this $\boldsymbol{p}$ if and only if there exists $\ell \in L$, $\ell + 1 \leq w \leq 2\ell$ such that $\boldsymbol{p} = \hat{\boldsymbol{p}}^l$ but $\sum_{t=\ell+1}^{w} a_{it} x_t(\hat{\boldsymbol{p}}^l) > \frac{l}{n} b_i$ for some i . Define $\delta = \frac{\epsilon}{m \cdot n^{m+1} \cdot E}$. Then by Hoeffding-Bernstein's Inequality one can show that for each fixed $\boldsymbol{p}$, i ℓ and w, the probability of that bad order is less than δ. Then by taking a union bound over all $\boldsymbol{p}$, i, ℓ and w, the lemma is proved.

A.6 Online Multi-dimensional Linear Program

Proof of Lemma 9

Using Lagrangian duality, observe that given optimal dual solution $\boldsymbol{p}^*$, optimal solution $\boldsymbol{x}^*$ is given by:

$$\begin{array}{ll} \text{maximize} & \boldsymbol{f}_t^T \boldsymbol{x}_t - \sum_i p_i^* \boldsymbol{g}_{it}^T \boldsymbol{x}_t \\ \text{subject to} & \boldsymbol{e}^T \boldsymbol{x}_t \leq 1, \boldsymbol{x}_t \geq 0. \end{array} \tag{A.7}$$

Therefore, it must be true that if $x_{tr}^* = 1$, then $r \in \arg\max_j f_{tj} - (\boldsymbol{p}^*)^T \boldsymbol{g}_{tj}$ and $f_{tr} - (\boldsymbol{p}^*)^T \boldsymbol{g}_{tr} \geq 0$. This means that for t's such that $\max_j f_{tj} - (\boldsymbol{p}^*)^T \boldsymbol{g}_{tj}$ is strictly positive and $\arg\max_j$ returns a unique solution, $\boldsymbol{x}_t(\boldsymbol{p}^*)$ and $\boldsymbol{x}_t^*$ are identical. By random perturbation argument there can be at most m values of t which do not satisfy this condition (for each such t, $\boldsymbol{p}$ satisfies an equation $f_{tj} - \boldsymbol{p}^T \boldsymbol{g}_{tj} = f_{tl} - \boldsymbol{p}^T \boldsymbol{g}_{tl}$ for some j, l, or $f_{tj} - \boldsymbol{p}^T \boldsymbol{g}_{tj} = 0$ for some j). This means $\boldsymbol{x}^*$ and $\boldsymbol{x}_t(\boldsymbol{p}^*)$ differ in at most m positions.

Proof of Lemma 10

Consider nk^2 expressions

$$\begin{array}{ll} f_{tj} - \boldsymbol{p}^T g_{tj} - (f_{tl} - \boldsymbol{p}^T g_{tl}), & 1 \leq j, l \leq k, j \neq l, 1 \leq t \leq n \\ f_{tj} - \boldsymbol{p}^T g_{tj}, 1 \leq j \leq k, & 1 \leq t \leq n. \end{array} \tag{A.8}$$

$\boldsymbol{x}_t(\boldsymbol{p})$ is completely determined once we determine the subset of expressions out of these nk^2 expressions that are assigned a non-negative value. By theory of computational geometry, there can be at most $(nk^2)^m$ such distinct assignments.

Appendix B

Supporting Proofs for Chapter 3

B.1 Examples of Demand Functions that Satisfy Assumption A

Example 1. (Linear Demand Function) For linear demand functions $\lambda(p) = a - bp$ with $0 < \underline{a} \leq a \leq \overline{a}$ and $0 < \underline{b} \leq b \leq \overline{b}$, all our assumptions hold with $M = \overline{a} - \underline{b}\underline{p}$, $K = \max\{\overline{b}, \underline{b}^{-1}, \overline{a} + 2\overline{b}\overline{p}\}$, $m_L = \overline{b}$ and $m_U = \underline{b}$.

Example 2. (Exponential Demand Function) For exponential demand functions $\lambda(p) = ae^{-bp}$ with $0 < \underline{a} \leq a \leq \overline{a}$ and $0 < \underline{b} \leq b \leq \overline{b}$, we have

- $|\lambda(p)| \leq \overline{a}e^{-\underline{b}\cdot\underline{p}}$
- $\lambda(p)$ is Lipschitz continuous with coefficient $\overline{a} \cdot \overline{b}$, $r(p)$ is Lipschitz continuous with coefficient $\overline{a} + \overline{a}\overline{b}\overline{p}$, and $\gamma(\lambda)$ is Lipschitz continuous with coefficient $\frac{e^{\overline{b}\cdot\overline{p}}}{\underline{a}\cdot\underline{b}}$
- $r(\lambda) = -\frac{\lambda}{b}\log\lambda + \frac{\lambda}{b}\log a$ is second-order differentiable and $-\frac{e^{\overline{b}\cdot\overline{p}}}{\underline{a}\cdot\underline{b}} \leq r''(\lambda) \leq -\frac{1}{\overline{a}\cdot\overline{b}}$.

Example 3. (Multi-nomial Logit Demand Function) For multi-nomiial logit demand functions $\lambda(p) = \frac{e^{a-bp}}{1+e^{a-bp}}$, with $0 < \underline{a} \leq a \leq \overline{a}$ and $0 < \underline{b} \leq b \leq \overline{b}$, we have

- $|\lambda(p)| \leq 1$

- $\lambda(p)$ is Lipschitz continuous with coefficient $\overline{b}$, $r(p)$ is Lipschitz continuous with coefficient $1+\overline{b}\cdot\overline{p}$, and $\gamma(\lambda)=\frac{1}{b}(\log\frac{1-\lambda}{\lambda}+a)$ is Lipschitz continuous with coefficient $\frac{e^{\overline{a}+\overline{b}\cdot\overline{p}}}{\underline{b}}$
- $r(\lambda)=\frac{\lambda}{b}(\log\frac{1-\lambda}{\lambda}+a)$ is second-order differentiable and $-\frac{1}{\underline{b}}e^{3(\overline{a}+\overline{b}\cdot\overline{p})}\leq r''(\lambda)\leq -\frac{4}{\overline{b}}$.

B.2 A Lemma on the Deviation of Poisson Random Variables

Lemma 26 *Suppose that $\mu\in[0,M]$ and $r_n\geq n^{\beta}$ with $\beta>0$. If*

$$\epsilon_n=2\eta^{1/2}M^{1/2}(\log n)^{1/2}r_n^{-1/2},$$

then for all $n\geq 1$,

$$P(N(\mu r_n)-\mu r_n>r_\epsilon n_n)\leq\frac{C}{n^\eta} \tag{B.1}$$

and

$$P(N(\mu r_n)-\mu r_n<-r_\epsilon n_n)\leq\frac{C}{n^\eta} \tag{B.2}$$

for some suitably chosen constant $C>0$.

We refer the online companion of [14] for the proof of this lemma.

B.3 Supporting Proofs for Section 3.5

Proof of Lemma 11

By plugging (3.29) and (3.31) into (3.24), we have

$$N^u=\min_l\{l: n^{\frac{1}{2}\cdot(\frac{3}{5})^l}<(\log n)^5\}. \tag{B.3}$$

Therefore, we can show that $N^u < \log n$ (for $n \geq 2$). Similarly, by plugging (3.32) and (3.34) into (3.28), we have

$$N^c = \min_l \{l : n^{\frac{1}{2}\cdot(\frac{2}{3})^l} < (\log n)^3\}.$$

Therefore, we can show that $N^c < \log n$ (for $n \geq 2$). Thus, our algorithm always stop in finite iterations and the number of iterations is bounded by $\log n$.

Proof of Lemma 12

Part 1: We first prove the first part, that is, when the algorithm runs within Step 2 until i reaches N^u.

We first show that if $p^D \in I_i^u$, then with probability $1 - O\left(\frac{1}{n^2}\right)$, $p^D \in I_{i+1}^u$. We assume $p^D \in I_i^u$. Now consider the next iteration. Define

$$u_n^i = 2\log n \cdot \max\left\{\left(\frac{\overline{p}_i^u - \underline{p}_i^u}{\kappa_i^u}\right)^2, \sqrt{\frac{\kappa_i^u}{n\tau_i^u}}\right\}.$$

First, we establish a bound on the difference in revenue $r(\lambda(p_i^u)) - r(\lambda(\hat{p}_i^u))$, where p_i^u is the revenue maximizing price on I_i^u and $\hat{p}_i^u$ is the empirical revenue maximizing price defined in our algorithm. We assume p_{i,j^*}^u is the nearest grid point to p_i^u in this iteration. We consider three cases:

- $p_i^u < p^u$: This is impossible since we know that $p^D \geq p^u > p_i^u$ and by the induction assumption $p^D \in I_i^u$. Therefore we must have $p^u \in I_i^u$, and by definition, p^u achieves a larger revenue rate than p_i^u, which is contradictory to the definition of p_i^u.

- $p_i^u = p^u$: In this case, by the granularity of the grid at iteration i, we have $|p_{i,j^*}^u - p^u| \leq \frac{\overline{p}_i^u - \underline{p}_i^u}{\kappa_i^u}$ and thus by our assumption that $r''(\lambda)$ is bounded, we know

that $|r(\lambda(p^u)) - r(\lambda(p^u_{i,j^*}))| \leq m_L K^2 \cdot (\frac{\overline{p}^u_i - \underline{p}^u_i}{\kappa^u_i})^2$, therefore we have:

$$\begin{aligned} & r(\lambda(p^u)) - r(\lambda(\hat{p}^u_i)) \\ = \; & r(\lambda(p^u)) - r(\lambda(p^u_{i,j^*})) + p^u_{i,j}\lambda(p^u_{i,j^*}) \\ & -p^u_{i,j^*}\hat{\lambda}(p^u_{i,j^*}) - (\hat{p}^u_i\lambda(\hat{p}^u_i) - \hat{p}^u_i\hat{\lambda}(\hat{p}^u_i)) + p^u_{i,j^*}\hat{\lambda}(p^u_{i,j^*}) - \hat{p}^u_i\hat{\lambda}(\hat{p}^u_i) \\ \leq \; & m_L K^2 (\tfrac{\overline{p}^u_i - \underline{p}^u_i}{\kappa^u_i})^2 + 2\max_{1\leq j\leq \kappa^u_i} |p^u_{i,j}\lambda(p^u_{i,j}) - p^u_{i,j}\hat{\lambda}(p^u_{i,j})|. \end{aligned} \tag{B.4}$$

In (B.4), $\hat{\lambda}$ is the observed demand rate and the last inequality is due to the definition of $\hat{p}^u_i$ and that $\hat{p}^u_i$ is among one of the $p^u_{i,j}$.

By Lemma 26 in Appendix B.2, we have

$$P(|\hat{\lambda}(p^u_{i,j}) - \lambda(p^u_{i,j})| > C\sqrt{\log n} \cdot \sqrt{\frac{\kappa^u_i}{n\tau^u_i}}) \leq \frac{1}{n^2}, \tag{B.5}$$

with some suitable constant C. Therefore, with probability $1-O(\frac{1}{n^2})$, $r(\lambda(p^u))-r(\lambda(\hat{p}^u_i)) \leq Cu^i_n$. However, by our assumption that $r''(\lambda) \leq m_U$ and that $\gamma(\lambda)$ is Lipschitz continuous, with probability $1 - O\left(\frac{1}{n^2}\right)$, $|p^u - \hat{p}^u_i| \leq C\sqrt{u^i_n}$ (here C represents some generic constant, depending only on the parameters in Γ and ϵ).

Now we consider the distance between $\hat{p}^c_i$ and p^c_i (this part of result can also be found in Lemma 4 in the online companion of [14]). Assume p_{i,j^*} is the nearest grid point to p^c_i. Then, using that we assumed $T = 1$, we have:

$$\begin{aligned} |\lambda(\hat{p}^c_i) - x| \; \leq \; & |\hat{\lambda}(\hat{p}^c_i) - x| + |\hat{\lambda}(\hat{p}^c_i) - \lambda(\hat{p}^c_i)| \\ \leq \; & |\hat{\lambda}(p^u_{i,j^*}) - x| + |\hat{\lambda}(\hat{p}^c_i) - \lambda(\hat{p}^c_i)| \\ \leq \; & |\lambda(p^u_{i,j^*}) - x| + |\hat{\lambda}(p^u_{i,j^*}) - \lambda(p^u_{i,j^*})| + |\hat{\lambda}(\hat{p}^c_i) - \lambda(\hat{p}^c_i)| \\ \leq \; & |\lambda(p^c_i) - x| + |\lambda(p^c_i) - \lambda(p^u_{i,j^*})| + 2\max_{1\leq j\leq \kappa^c_i} |\hat{\lambda}(p^u_{i,j}) - \lambda(p^u_{i,j})|. \end{aligned} \tag{B.6}$$

By the definition of p^c_i, $\lambda(p^c_i)-x$ and $\lambda(\hat{p}^c_i)-x$ must have the same sign, otherwise there exists a point in between that achieves a smaller value of $|\lambda(p) - x|$.

Therefore we have

$$|\lambda(p_i^c) - \lambda(\hat{p}_i^c)| \leq |\lambda(p_i^c) - \lambda(p_{i,j^*}^u)| + 2 \max_{1 \leq j \leq \kappa_i^c} |\hat{\lambda}(p_{i,j}^u) - \lambda(p_{i,j}^u)|. \tag{B.7}$$

By the Lipschitz continuity of λ, we have

$$|\lambda(p_i^c) - \lambda(p_{i,j^*}^u)| \leq K \frac{\overline{p}_i^u - \underline{p}_i^u}{\kappa_i^u}. \tag{B.8}$$

Also by Lemma 26 in Appendix B.2, we have with probability $1 - O\left(\frac{1}{n^2}\right)$,

$$\max_{1 \leq j \leq \kappa_i^c} |\hat{\lambda}(p_{i,j}^u) - \lambda(p_{i,j}^u)| \leq C\sqrt{\log n} \cdot \sqrt{\frac{\kappa_i^u}{n\tau_i^u}}. \tag{B.9}$$

Therefore, with probability $1 - O\left(\frac{1}{n^2}\right)$, we have

$$|\lambda(p_i^c) - \lambda(\hat{p}_i^c)| \leq C\sqrt{u_n^i} \tag{B.10}$$

and by the Lipschitz continuity of $\nu(\lambda)$, this implies that with probability $1 - O\left(\frac{1}{n^2}\right)$,

$$|\hat{p}_i^c - p_i^c| \leq C\sqrt{u_n^i}. \tag{B.11}$$

Therefore, we have

$$\begin{aligned} & P\{|\hat{p}_i - p^D| > C\sqrt{u_n^i}\} \\ \leq \; & P\{|\hat{p}_i^c - p_i^c| > C\sqrt{u_n^i}\} + P\{|\hat{p}_i^u - p^u| > C\sqrt{u_n^i}\} \leq O\left(\frac{1}{n^2}\right). \end{aligned} \tag{B.12}$$

Here we used the fact that:

$$|\max\{a, c\} - \max\{b, d\}| > u \quad \Rightarrow |a - b| > u \text{ or } |c - d| > u.$$

Note that (B.12) is equivalent to saying that

$$P(p^D \in [\hat{p}_i - C\sqrt{u_n^i}, \hat{p}_i + C\sqrt{u_n^i}]) > 1 - O\left(\frac{1}{n^2}\right). \tag{B.13}$$

Now also note that the interval I_{i+1} in our algorithm is chosen to be

$$\left[\hat{p}_i - \frac{\log n}{3}\frac{\overline{p}_i^u - \underline{p}_i^u}{\kappa_i^u}, \hat{p}_i + \frac{2\log n}{3}\frac{\overline{p}_i^u - \underline{p}_i^u}{\kappa_i^u}\right],$$

which is of order $\sqrt{\log n}$ greater than $\sqrt{u_n^i}$ (and according to the way we defined κ_i^u and τ_i^u, the two terms in u_n^i are of the same order). Therefore we know that with probability $1 - O(\frac{1}{n^2})$, $p^D \in I_{i+1}^u$.

- $p^u < p_i^u$: In this case, $p^D = p^c$. With the same argument, but only the p^c part, we know that with probability $1 - O\left(\frac{1}{n^2}\right)$, $p^D \in I_{i+1}^u$.

Also, as claimed in the previous lemma, the number of steps N^u is less than $\log n$. Therefore, we can take a union bound over N^u steps, and claim that with probability $1 - O\left(\frac{1}{n}\right)$, $p^D \in I_i^u$, for all $i = 1, ..., N^u$.

Part 2: Using the same argument as in Part 1, we know that with probability $1 - O\left(\frac{1}{n}\right)$, $p^D \in I_i^u$ for $i = 1, ..., i_0$. Now at i_0, condition (3.16) is triggered. We first claim that whenever (3.16) is satisfied, with probability $1 - O\left(\frac{1}{n}\right)$, $p^D = p^c$.

By the argument in Part 1, we know that with probability $1 - O\left(\frac{1}{n}\right)$,

$$|\hat{p}_{i_0}^c - p_{i_0}^c| \leq \sqrt{\log n} \cdot \frac{\overline{p}_{i_0}^u - \underline{p}_{i_0}^u}{\kappa_{i_0}^u},$$

$$|\hat{p}_{i_0}^u - p_{i_0}^u| \leq \sqrt{\log n} \cdot \frac{\overline{p}_{i_0}^u - \underline{p}_{i_0}^u}{\kappa_{i_0}^u}.$$

Therefore, if

$$\hat{p}_{i_0}^c > \hat{p}_{i_0}^u + 2\sqrt{\log n} \cdot \frac{\overline{p}_{i_0}^u - \underline{p}_{i_0}^u}{\kappa_{i_0}^u} \tag{B.14}$$

holds, then with probability $1 - O\left(\frac{1}{n}\right)$,

$$p^c_{i_0} > p^u_{i_0}.$$

And when (B.14) holds, p^c_i is not the left end-point of $I^u_{i_0}$ and $p^u_{i_0}$ is not the right end-point of $I^u_{i_0}$, which means

$$p^u \leq p^u_{i_0} < p^c_{i_0} \leq p^c = p^D.$$

Now we consider the procedure in Step 3 of our algorithm and show that with probability $1 - O\left(\frac{1}{n}\right)$, $p^D = p^c \in I^c_i$ for all $i = 1, 2, ..., N^c$.

We again prove that if $p^D \in I^c_i$, then with probability $1 - O\left(\frac{1}{n^2}\right)$, $p^D = p^c \in I^c_{i+1}$. We consider $\hat{q}^c_i$ (which is the optimal empirical solution in Step 3 in our algorithm). Define in this case

$$v^i_n = 2\sqrt{\log n} \cdot \max\left\{\frac{\overline{p}^c_i - \underline{p}^c_i}{\kappa^c_i}, \sqrt{\frac{\kappa^c_i}{n\tau^c_i}}\right\}.$$

Using the same discussion as in Part 1, we have with probability $1 - O\left(\frac{1}{n^2}\right)$,

$$|\hat{q}_i - p^c_i| = |\hat{p}^c_i - p^c| \leq C v^i_n.$$

However, remember that in our algorithm, the next interval is defined to be

$$I^c_{i+1} = \left[\hat{q}_i - \frac{\log n}{2} \cdot \frac{\overline{p}^c_i - \underline{p}^c_i}{\kappa^c_i}, \hat{q}_i + \frac{\log n}{2} \cdot \frac{\overline{p}^c_i - \underline{p}^c_i}{\kappa^c_i}\right],$$

which is of order $\sqrt{\log n}$ larger than v^i_n. Therefore with probability $1 - O\left(\frac{1}{n^2}\right)$, $p^D = p^c \in I^c_{i+1}$. Again, taking a union bound over these N^c steps (no more than $\log n$) results in this lemma.

Proof of Lemma 14

Proof. Define $A_{ij}^u = \{\omega : Y_{ij}^u - EY_{ij}^u \leq EY_{ij}^u\}$. First we show that

$$\bigcap_{i,j} A_{ij}^u \subset A_1^u. \tag{B.15}$$

To show this we only need to show that $2\sum_{i,j} EY_{ij}^u \leq nx$. Recall that $|\lambda(p)| \leq M$, for all $p \in [\underline{p}, \overline{p}]$. We have:

$$\sum_{i,j} EY_{ij}^u \leq M \sum_{i,j} n\Delta_i = Mn \sum_{i=1}^{N^u} \tau_i^u \leq MnN^u \tau_{N^u}^u. \tag{B.16}$$

By our definition of τ_i^u, we know that $\tau_{N^u}^u$ is of order less than 1 (otherwise the algorithm is stopped at the previous iteration). Therefore, when n is large enough, $2\sum_{i,j} EY_{i,j}^u \leq nx$, i.e., (B.15) holds uniformly in n.

However, by Lemma 26, we know that $P(A_{ij}^u) \geq 1 - O\left(\frac{1}{n^3}\right)$ and thus

$$P(A_1) \geq 1 - O\left(\frac{1}{n}\right),$$

since each $\kappa_i^u \leq n$ and $N^u \leq \log n$.

Now we can rewrite the left-hand side of (3.37) as:

$$\begin{aligned}
& E[\sum_{i=1}^{N^u} \sum_{j=1}^{\kappa_i^u} p_{i,j}^u Y_{ij}^u I(A_1^u) I(A_2^u)] \\
= & \sum_{i=1}^{N^u} \sum_{j=1}^{\kappa_i^u} E[p_{i,j}^u Y_{ij}^u I(A_1^u) I(A_2^u)] \\
= & \sum_{i=1}^{N^u} \sum_{j=1}^{\kappa_i^u} E[p_{i,j}^u I(A_1^u) I(A_2^u) E[Y_{ij}^u I(A_1^u A_2^u)]] \\
= & \sum_{i=1}^{N^u} \sum_{j=1}^{\kappa_i^u} E[p_{i,j}^u I(A_1^u) I(A_2^u) (E[Y_{ij}^u] - E[Y_{ij}^u I((A_1^u)^c \cup (A_2^u)^c)])].
\end{aligned} \tag{B.17}$$

However, by Cauchy-Schwartz inequality and the property of Poisson distribution $(EN(\lambda) = \lambda, E[N(\lambda)^2] = \lambda^2 + \lambda)$

$$E[Y_{ij}^u I((A_1^u)^c \cup (A_2^u)^c)] \leq \sqrt{[E(Y_{ij}^u)]^2 \cdot P((A_1^u)^c \cup (A_2^u)^c)} \leq O\left(\frac{1}{\sqrt{n}}\right) E[Y_{ij}^u]. \quad \text{(B.18)}$$

Now we plug this back into (B.17) and obtain

$$\begin{aligned}
& E[\sum_{i=1}^{N^u}\sum_{j=1}^{\kappa_i^u} p_{i,j}^u Y_{ij}^u I(A_1^u) I(A_2^u)] \\
\geq\ & (1 - O\left(\frac{1}{\sqrt{n}}\right)) \sum_{i=1}^{N^u}\sum_{j=1}^{\kappa_i^u} E[p_{i,j}^u E Y_{ij}^u I(A_1^u) I(A_2^u)] \\
=\ & (1 - O\left(\frac{1}{\sqrt{n}}\right)) \sum_{i=1}^{N^u}\sum_{j=1}^{\kappa_i^u} E[p_{i,j}^u \lambda(p_{i,j}^u) n \Delta_i^u I(A_1^u) I(A_2^u)] \\
=\ & (1 - O\left(\frac{1}{\sqrt{n}}\right)) \sum_{i=1}^{N^u}\sum_{j=1}^{\kappa_i^u} E[p_{i,j}^u \lambda(p_{i,j}^u) n \Delta_i^u | I(A_1^u) I(A_2^u)] P(A_1^u A_2^u) \\
=\ & (1 - O\left(\frac{1}{\sqrt{n}}\right)) \sum_{i=1}^{N^u}\sum_{j=1}^{\kappa_i^u} E[p_{i,j}^u \lambda(p_{i,j}^u) n \Delta_i^u | I(A_1^u) I(A_2^u)].
\end{aligned} \quad \text{(B.19)}$$

Now we consider

$$\sum_{i=1}^{N^u}\sum_{j=1}^{\kappa_i^u} E[p_{i,j}^u \lambda(p_{i,j}^u) n \Delta_i^u | I(A_1^u) I(A_2^u)]. \quad \text{(B.20)}$$

By the bound on the second derivative of the function $r(\lambda)$ and the assumption that $p^D = p^u$:

$$p_{i,j}^u \lambda(p_{i,j}^u) = r(\lambda(p_{i,j}^u)) \geq r(\lambda(p^D)) - m_L(\lambda(p_{i,j}^u) - \lambda(p^D))^2 \geq p^D \lambda(p^D) - m_L K^2 (\overline{p}_i^u - \underline{p}_i^u)^2. \quad \text{(B.21)}$$

Therefore, we have

$$\begin{aligned}
& (1-O\left(\frac{1}{\sqrt{n}}\right))\cdot\sum_{i=1}^{N^u}\sum_{j=1}^{\kappa_i^u}E[p_{i,j}^u\lambda(p_{i,j}^u)n\Delta_i|I(A_1^u)I(A_2^u)] \\
\geq\ & (1-O\left(\frac{1}{\sqrt{n}}\right))\cdot(\sum_{i=1}^{N^u}p^D\lambda(p^D)n\tau_i^u-m_LK^2\sum_{i=1}^{N^u}(\overline{p}_i^u-\underline{p}_i^u)^2n\tau_i^u) \\
\geq\ & (1-O\left(\frac{1}{\sqrt{n}}\right))\cdot(\sum_{i=1}^{N^u}p^D\lambda(p^D)n\tau_i^u-m_LK^2N^un\tau_1^u) \\
\geq\ & \sum_{i=1}^{N^u}p^D\lambda(p^D)n\tau_i^u-Cn\tau_1^u\log n,
\end{aligned} \tag{B.22}$$

where the second to last step is due to (3.22) and (3.23), and the last step is because $\frac{1}{\sqrt{n}}\tau_i^u=n^{-\frac{1}{2}-\frac{1}{2}(3/5)^{i-1}}(\log n)^5<\tau_1^u$, for $i=1,...,N^u$. Therefore, the lemma holds.

Proof of Lemma 15 and 16

Proof of Lemma 15. First we show that with probability $1-O(\frac{1}{n})$,

$$\sum_{i,j}Y_{ij}^u+\hat{Y}^u-\sum_{i,j}EY_{ij}^u-E\hat{Y}^u\leq Cn\tau_1^u\log n. \tag{B.23}$$

Then by Cauchy-Schwartz inequality,

$$\begin{aligned}
E[(\hat{Y}^u+\sum_{i,j}Y_{ij}^u-E\hat{Y}^u-\sum_{i,j}EY_{ij}^u)^+ \quad & I(\hat{Y}^u+\textstyle\sum_{i,j}Y_{ij}^u-E\hat{Y}^u-\sum_{i,j}EY_{ij}^u>Cn\tau_1^u\log n)] \\
& \leq O\left(\tfrac{1}{\sqrt{n}}\right)(E\hat{Y}^u+\textstyle\sum_{i,j}EY_{ij}^u)\leq Cn\tau_1^u\log n,
\end{aligned}$$

where the second to last inequality is because that for a Poisson random variable $N(\mu)$, $Var(N(\mu))=\mu^2$ and the last inequality is because the demand rate is bounded and $\tau_1^u\geq n^{-1/2}$. Therefore, we have

$$E(\hat{Y}^u+\sum_{i,j}Yij^u-E\hat{Y}^u-\sum_{i,j}EY_{ij}^u)^+\leq Cn\tau_1^u\log n, \tag{B.24}$$

which implies our lemma.

To show (B.23), we apply Lemma 26. For each given i, j, by Lemma 26, we have

$$P(Y_{ij}^u - EY_{ij}^u > 3M\sqrt{n\Delta_i^u \log n}) \leq \frac{C}{n^3}.$$

By taking a union bound over all i, j, we get

$$\begin{aligned} & P(\sum_{i,j} Y_{ij}^u - \sum_{i,j} EY_{ij}^u > 3M \sum_i \kappa_i^u \sqrt{n\Delta_i^u \log n}) \\ \leq & \sum_{i=1}^{N^u} \sum_{j=1}^{\kappa_i^u} P(Y_{ij}^u - EY_{ij}^u > 3M\sqrt{n\Delta_i^u \log n}) \leq O\left(\frac{1}{n}\right), \end{aligned}$$

where the last step is because $\kappa_i^u < n$ and $N^u \leq \log n$.

On the other hand, by the definition of τ_i^u and κ_i^u, we have

$$3M \sum_i \kappa_i^u \sqrt{n\Delta_i^u \log n} = 3M\sqrt{n \log n} \sum_{i=1}^{N^u} \sqrt{\kappa_i^u \tau_i^u} \leq CN^u n\tau_1^u \leq Cn\tau_1^u \log n, \quad \text{(B.25)}$$

where the last inequality follows from the definition of κ_i^u and τ_i^u in (3.29) and (3.30). We then consider $\hat{Y}^u - E[\hat{Y}^u]$, again use inequality (B.1). We have

$$P(\hat{Y}^u - E[\hat{Y}^u] > 2M\sqrt{n \log n}) < \frac{C}{n^2},$$

since $\sqrt{n \log n} < n\tau_1^u$, the lemma holds. □

Proof of Lemma 16. By definition, we have

$$EY_{ij}^u = n\lambda(p_{i,j}^u)\Delta_i^u,$$

where

$$p_{i,j}^u = \underline{p}_i^u + (j-1)\frac{\overline{p}_i^u - \underline{p}_i^u}{\kappa_i^u} = \hat{p}_{i-1} - \frac{\log n}{3} \cdot \frac{\overline{p}_{i-1}^u - \underline{p}_{i-1}^u}{\kappa_{i-1}^u} + (j-1)\frac{\overline{p}_i^u - \underline{p}_i^u}{\kappa_i^u}.$$

By our above discussion, with probability $1 - O\left(\frac{1}{n}\right)$, condition (3.16) doesn't hold, i.e., $\hat{p}_{i-1} \geq \hat{p}^c_{i-1} - 2\sqrt{\log n} \cdot \frac{\overline{p}^u_{i-1} - \underline{p}^u_{i-1}}{\kappa^u_{i-1}}$. And as we showed in (B.11), $\hat{p}^c_{i-1} \geq p^c_{i-1} - \sqrt{\log n} \cdot \frac{\overline{p}^u_{i-1} - \underline{p}^u_{i-1}}{\kappa^u_{i-1}}$. Also since $p^u \geq p^c$, we must have $p^c_{i-1} \geq p^c$. Therefore we have,

$$\begin{aligned} p^u_{i,j} &\geq p^c + (j-1)\frac{\overline{p}^u_i - \underline{p}^u_i}{\kappa^u_i} - \frac{\log n}{3} \cdot \frac{\overline{p}^u_{i-1} - \underline{p}^u_{i-1}}{\kappa^u_{i-1}} - 3\sqrt{\log n} \cdot \frac{\overline{p}^u_{i-1} - \underline{p}^u_{i-1}}{\kappa^u_{i-1}} \\ &\geq p^c + (j-1)\frac{\overline{p}^u_i - \underline{p}^u_i}{\kappa^u_i} - \frac{\log n}{2} \cdot \frac{\overline{p}^u_{i-1} - \underline{p}^u_{i-1}}{\kappa^u_{i-1}}, \end{aligned} \tag{B.26}$$

when $\sqrt{\log n} \geq 6$. Using the Taylor expansion for $\lambda(p)$, we have that

$$\begin{aligned} \lambda(p^u_{i,j}) &\leq \lambda(p^c) + ((j-1)\frac{\overline{p}^u_i - \underline{p}^u_i}{\kappa^u_i} - \frac{\log n}{2} \cdot \frac{\overline{p}^u_{i-1} - \underline{p}^u_{i-1}}{\kappa^u_{i-1}})\lambda'(p^c) + \sup\{\lambda''(p^c)\}(\overline{p}^u_i - \underline{p}^u_i)^2 \\ &\leq \lambda(p^c) + C \cdot (\overline{p}^u_i - \underline{p}^u_i)^2. \end{aligned} \tag{B.27}$$

The last inequality uses the fact that $\lambda''(p)$ is bounded by a constant which is not hard to derive from Assumption A. Therefore

$$\sum_{i,j} EY^u_{ij} \leq n\lambda(p^c)t^u_{N^u} + Cn\sum_{i=1}^{N^u} \tau^u_i(\overline{p}^u_i - \underline{p}^u_i)^2 \leq n\lambda(p^c)t^u_{N^u} + Cn\tau^u_1 \log n, \tag{B.28}$$

where the last equation follows from (3.22) and (3.23).

Also we have

$$E\hat{Y}^u = \lambda(\tilde{p})n(1 - t^u_{N^u}),$$

and with probability $1 - O\left(\frac{1}{n}\right)$,

$$\begin{aligned} \tilde{p} &= \hat{p}_{N^u} + 2\sqrt{\log n} \cdot \frac{\overline{p}^u_{N^u} - \underline{p}^u_{N^u}}{\kappa^{N^u}_i} \\ &\geq \max(\hat{p}^u_{N^u}, \hat{p}^c_{N^u}) + 2\sqrt{\log n} \cdot \frac{\overline{p}^u_{N^u} - \underline{p}^u_{N^u}}{\kappa^{N^u}_i} \\ &\geq \max(p^u_{N^u}, p^c_{N^u}) \\ &\geq p^D \geq p^c, \end{aligned} \tag{B.29}$$

where the first equation is due to the definition of $\tilde{p}$, the second one is due to (B.12), and the last one is by Lemma 12. Therefore,

$$E\hat{Y}^u \leq \lambda(p^c)n(1-t^u_{N^u})$$

and thus

$$\sum_{i,j} E\hat{Y}_{ij} + E\hat{Y}^u \leq nx + Cn\tau^u_1 \log n. \qquad \square$$

Proof of Lemma 17, 18, 19, and 20

Proof of Lemma 17. Define $A_1 = \{\omega : p^D \in I^u_i \text{ for all } i\}$. By Lemma 12, we know that $P(A_1) \geq 1 - O\left(\frac{1}{n}\right)$, and we have

$$E[\sum_{i=1}^{N^u}\sum_{j=1}^{\kappa^u_i} p^u_{i,j}Y^u_{ij}I(\cup_{l=i}^{N^u+1}B_l)] \geq \sum_{i=1}^{N^u}\sum_{j=1}^{\kappa^u_i} E[p^u_{i,j}E[Y^u_{ij}I(\cup_{l=i}^{N^u+1}B_l)]\cdot I(\cup_{l=1}^{N^u+1}B_l)I(A_1)]. \tag{B.30}$$

However, note that $\cup_{l=i}^{N^u+1}B_l = (\cup_{l=1}^{i-1}B_l)^c$ only depends on the realization up to period $i-1$. Therefore, we know that Y^u_{ij} given $p^u_{i,j}$ is independent of $\cup_{l=i}^{N^u+1}B_l$. Therefore, we have

$$\begin{aligned}
E[\sum_{i=1}^{N^u}\sum_{j=1}^{\kappa^u_i} p^u_{i,j}Y^u_{ij}I(\cup_{l=i}^{N^u+1}B_l)] &\geq \sum_{i=1}^{N^u}\sum_{j=1}^{\kappa^u_i} E[n\Delta^u_i p^u_{i,j}\lambda(p^u_{i,j})I(\cup_{l=i}^{N^u+1}B_l)I(A_1)] \\
&\geq (1-O\left(\frac{1}{n}\right))\sum_{i=1}^{N^u}\sum_{j=1}^{\kappa^u_i} E[n\Delta^u_i p^u_{i,j}\lambda(p^u_{i,j})I(\cup_{l=i}^{N^u+1}B_l)|I(A_1)] \\
&\geq (1-O\left(\frac{1}{n}\right))\sum_{i=1}^{N^u}\Big(n\tau^u_i p^D\lambda(p^D)P(\cup_{l=i}^{N^u+1}B_l) \\
&\quad -n\tau^u_i m_L K^2(\frac{\overline{p}^u_i - \underline{p}^u_i}{\kappa^u_i})^2\Big) \\
&\geq \sum_{i=1}^{N^u} n\tau_i p^D\lambda(p^D)P(\cup_{l=i}^{N^u+1}B_l) - Cn\tau_1\log n, \qquad (B.31)
\end{aligned}$$

where the second to last inequality is because of the bounded second derivative of the

revenue function and that A holds; and the last inequality is because of the relation of (3.23) and that $N^u \leq \log n$. $\square$

Proof of Lemma 18. Define

$$A_2 = \{\omega : \text{For each } i, j, Y_{ij}^u \leq EY_{ij}^u + 3M\sqrt{n\Delta_i^u \log n} \text{ and } \hat{Y}^u \leq E\hat{Y}^u + 3M\sqrt{n \log n}\}. \tag{B.32}$$

By Lemma 26, $P(A_2) = 1 - O\left(\frac{1}{n}\right)$. Similar to (3.38), we have the following relation:

$$\begin{aligned} & E[\tilde{p}\min(\hat{Y}^u, (nx - \sum_{i,j} Y_{ij}^u)^+) I(A_2) I(B_{N^u+1})] \\ \geq \; & E[\tilde{p}\hat{Y}^u I(A_2) I(B_{N^u+1})] - E[\tilde{p}(\hat{Y}^u + \sum_{i,j} Y_{ij}^u - nx)^+ I(A_2) I(B_{N^u+1})]. \end{aligned} \tag{B.33}$$

And by the same argument as (3.41) we have

$$\begin{aligned} E[\tilde{p}\hat{Y}^u I(A_2) I(B_{N^u+1})] &\geq (1 - O\left(\frac{1}{n}\right))(p^D \lambda(p^D) \cdot n(1 - t_{N^u}^u) P(B_{N^u+1})) - Cn\tau_1^u) \\ &\geq p^D \lambda(p^D) \cdot n(1 - t_{N^u}^u) P(B_{N^u+1})) - Cn\tau_1^u. \end{aligned} \tag{B.34}$$

Now we consider

$$E[\tilde{p}(\hat{Y}^u + \sum_{i,j} Y_{ij}^u - nx)^+ I(A_2) I(B_{N^u+1})]. \tag{B.35}$$

We first relax it to

$$\bar{p}E[(\hat{Y}^u + \sum_{i,j} Y_{ij}^u - nx)^+ I(A_2) I(B_{N^u+1})]. \tag{B.36}$$

Conditional on A, $Y_{ij}^u \leq EY_{ij}^u + 3M\sqrt{n\Delta_i^u \log n}$ for all i, j and $\hat{Y}^u \leq EY_{ij}^u + 3M\sqrt{n \log n}$, and by the argument in Lemma 15, we have $\sum_{i,j} 3M\sqrt{n\Delta_i^u \log n} + 2M\sqrt{n \log n} \leq Cn\tau_1^u \log n$. Also, by the same argument as in Lemma 16, we have

$\sum_{i,j} EY_{ij}^u + E\hat{Y}^u - nx \leq Cn\tau_1^u \log n$. Therefore, the lemma holds. □

Proof of Lemma 19. Define $A_3 = \{\omega : p^D \in I_i^c, \forall i\}$. By Lemma 12, we know that $P(A_3) = 1 - O\left(\frac{1}{n}\right)$. Also note that each Y_{ij}^c given $p_{i,j}^c$ is independent with $\cup_{l=1}^{N^u} B_l$. Therefore we have,

$$\begin{aligned}
& \sum_{i=1}^{N^c} \sum_{j=1}^{\kappa_i^c} E[p_{i,j}^c Y_{ij}^c I(\cup_{l=1}^{N^u} B_l)] \\
\geq & \sum_{i=1}^{N^c} \sum_{j=1}^{\kappa_i^c} E[p_{i,j}^c E[Y_{ij}^c I(\cup_{l=1}^{N^u} B_l)] I(\cup_{l=1}^{N^u} B_l) I(A_3)] \\
= & \sum_{i=1}^{N^c} \sum_{j=1}^{\kappa_i^c} E[n\Delta_i^c p_{i,j}^c \lambda(p_{i,j}^c) I(\cup_{l=1}^{N^u} B_l) I(A_3)] \\
\geq & (1 - O\left(\frac{1}{n}\right)) \left(\sum_{i=1}^{N^c} n\tau_i^c p^D \lambda(p^D) P(\cup_{l=1}^{N^u} B_l) - \sum_{i=1}^{N^c} M(\overline{p}_i^c - \underline{p}_i^c)\tau_i^c n \right) \\
\geq & \sum_{i=1}^{N^c} n\tau_i^c p^D \lambda(p^D) P(\cup_{l=1}^{N^u} B_l) - Cn\tau_1^c \log n,
\end{aligned} \tag{B.37}$$

where the equality is because of the independence of Y_{ij}^c and B_l as we argued above, the second inequality is because $p^D \in I_i^c$ for all i and the Lipschitz continuity of the revenue rate function. The last inequality is because of the relation (3.27) and that N^c is bounded by $\log n$. □

Proof of Lemma 20. Define

$$\begin{aligned}
A_4 = \{\omega : Y_{ij}^u \leq EY_{ij}^u + 3M\sqrt{n\Delta_i^u \log n} \quad & \text{and } Y_{ij}^c \leq EY_{ij}^c + 3M\sqrt{n\Delta_i^c \log n}, \forall i, j; \\
& \hat{Y}_i^c \leq E\hat{Y}_i^c + 3M\sqrt{n \log n} \text{ and } p^D \in I_i^c, \forall i\}.
\end{aligned} \tag{B.38}$$

By Lemma 26, we have that $P(A_4) = 1 - O\left(\frac{1}{n}\right)$.

By the same transformation as in (B.33), we have

$$\begin{aligned} & E[\tilde{q}\min(\hat{Y}_l^c, (nx - \sum_{i=1}^{l}\sum_{j=1}^{\kappa_i^u} Y_{ij}^u - \sum_{i=1}^{N^c}\sum_{j=1}^{\kappa_i^c} Y_{ij}^c)^+)I(B_l)] \\ \geq \;& E[\tilde{q}\hat{Y}_l^c I(B_l)I(A_4)] - E[\tilde{q}(\sum_{i,j} Y_{ij}^c + \sum_{i=1}^{l}\sum_{j=1}^{\kappa_i^u} Y_{ij}^u + \hat{Y}_l^c - nx)^+ I(B_l)I(A_4)]. \end{aligned} \tag{B.39}$$

For the first term, we have

$$\begin{aligned} & E[\tilde{q}\hat{Y}_l^c I(B_l)I(A_4)] \\ \geq \;& (1 - O\left(\frac{1}{n}\right))E(\tilde{q}E[\hat{Y}_l^c]I(B_l)|I(A_4)) \\ \geq \;& (1 - O\left(\frac{1}{n}\right))E(n(1 - t_l^u - t_{N^c}^c)\tilde{q}\lambda(\tilde{q})I(B_l)) \\ \geq \;& (1 - O\left(\frac{1}{n}\right))(n(1 - t_l^u - t_{N^c}^c)p^D\lambda(p^D)P(B_l) - n(1 - t_l^u - t_{N^c}^c)(\overline{p}_{N^c+1}^c - \underline{p}_{N^c+1}^c)) \\ \geq \;& n(1 - t_l^u - t_{N^c}^c)p^D\lambda(p^D)P(B_l) - Cn\tau_1^c, \end{aligned} \tag{B.40}$$

where the third inequality is because of the definition of $\tilde{q}$ and the Lipschitz continuity of the revenue rate function. And the last inequality is due to condition (3.28).

For the second term, we first relax it to

$$\bar{p}E[(\sum_{i,j} Y_{ij}^c + \sum_{i=1}^{l}\sum_{j=1}^{\kappa_i^u} Y_{ij}^u + \hat{Y}_l^c - nx)^+ I(B_l)I(A_4)].$$

By the definition of A_4, we have,

$$\begin{aligned} & E[(\sum_{i,j} Y_{ij}^c + \sum_{i=1}^{l}\sum_{j=1}^{\kappa_i^u} Y_{ij}^u + \hat{Y}_l^c - nx)^+ I(B_l) I(A_4)] \\ \leq \; & (\sum_{i,j} EY_{ij}^c + \sum_{i=1}^{l}\sum_{j=1}^{\kappa_i^u} EY_{ij}^u + E\hat{Y}_l^c - nx)^+ + Cn\tau_1^c \log n \leq Cn\tau_1^c \log n. \end{aligned} \tag{B.41}$$

The first inequality is because as we argued in Lemma 15, $\sum_{i,j} 3M\sqrt{n\Delta_i^u \log n} + \sum_{i,j} 3M\sqrt{n\Delta_j^c \log n} + M\sqrt{n \log n} \leq Cn\tau_1^u \log n$, and the second inequality is because

$$\begin{aligned} & \sum_{i,j} EY_{ij}^c + \sum_{i=1}^{l}\sum_{j=1}^{\kappa_i^u} EY_{ij}^u + E\hat{Y}_l^c \\ \leq \; & \sum_{i=1}^{l} n\tau_i^u \lambda(p_{i,j}^u) + \sum_{i=1}^{N^c} n\tau_i^c \lambda(p_{i,j}^c) + n(1 - t_l^u - t_{N^c}^c)\lambda(\tilde{q}) \\ \leq \; & \sum_{i=1}^{l} n\tau_i^u \lambda(p^D) + C_1 n\tau_1^u \log n \\ & + \sum_{i=1}^{N^c} n\tau_i^c \lambda(p^D) + C_2 n\tau_1^c \log n + n(1 - t_l^u - t_{N^c}^c)\lambda(p^D) + C_3 n\tau_1^c \\ \leq \; & nx + Cn\tau_1^u \log n, \end{aligned} \tag{B.42}$$

where the first inequality is by definition, the first part of the second inequality is by Lemma 16, the second and third part of the second inequality is due to the continuity of the demand rate function and that $p^D \in I_i^c$, so that $p_{i,j}^c - p^D \leq M(\overline{p}_i^c - \underline{p}_i^c)$ and the relation in (3.27). And the last inequality is because when $p^c > p^u$, $\lambda(p^D)$ is x (remember that we assumed $T = 1$).

Combining (B.40) and (B.42) together shows that the lemma holds. □

B.4 Supporting Proofs for Section 3.6

Proof of Lemma 22

Consider the final term in (3.54). Note that we have the following simple inequality:

$$\log(x+1) \geq x - \frac{x^2}{2(1-|x|)}, \quad \forall x < 1.$$

Therefore, we have

$$-\log x - 1 \leq -x + \frac{(x-1)^2}{2(2-x)} \quad \forall\ 0 < x < 2.$$

Apply this relationship to the final term in (3.54) and note that for any $z \in [1/3, 2/3]$ and $p(s) \in [1/2, 3/2]$,

$$\frac{2}{3} \leq \frac{\frac{1}{2} + z - zp(s)}{\frac{1}{2} + z_0 - z_0 p(s)} \leq 2, \tag{B.43}$$

we have

$$\begin{aligned} \mathcal{K}(\mathcal{P}_{z_0}^{\pi}, \mathcal{P}_{z}^{\pi}) &\leq nE_{z_0}^{\pi} \int_0^1 \frac{1}{2(2 - \frac{1/2+z_0-z_0p(s)}{1/2+z-zp(s)})} \frac{(z_0-z)^2(1-p(s))^2}{(1/2+z_0-z_0p(s))^2} ds \\ &\leq nE_{z_0}^{\pi} \int_0^1 \frac{(z_0-z)^2(1-p(s))^2}{(1/2+z_0-z_0p(s))^2} ds. \end{aligned} \tag{B.44}$$

Also, for $z \in [1/3, 2/3]$ and $p(s) \in [1/2, 3/2]$, we have

$$\frac{1}{2} + z_0 - z_0 p(s) \geq \frac{1}{4}. \tag{B.45}$$

Therefore, we have

$$\mathcal{K}(\mathcal{P}_{z_0}^{\pi}, \mathcal{P}_{z}^{\pi}) \leq 16n(z_0-z)^2 E_{z_0}^{\pi} \int_0^1 (1-p(s))^2 ds.$$

However, under the case when the parameter is z_0, we have $p^D = 1$ and

$$\begin{aligned} R_n^\pi(x,T;z_0) &= 1 - \frac{J_n^\pi(x,T;z_0)}{J_n^D(x,T;z_0)} \\ &\geq \frac{E_{z_0}^\pi \int_0^1 (r(p^D) - r(p(s)))ds}{E_{z_0}^\pi \int_0^1 r(p^D)ds} \\ &\geq \frac{2}{3} E_{z_0}^\pi \int_0^1 (1-p(s))^2 ds, \end{aligned} \tag{B.46}$$

where the first inequality follows from the definition of J^D and that we relaxed the inventory constraint, and the second inequality is because of the 4th condition in Lemma 21. Therefore,

$$\mathcal{K}(\mathcal{P}_{z_0}^\pi, \mathcal{P}_z^\pi) \leq 24n(z_0 - z)^2 R_n^\pi(x,T;z_0). \tag{B.47}$$

Proof of Lemma 23

We define two intervals C_{z_0} and $C_{z_1^n}$ as follows:

$$C_{z_0} = \left[p^D(z_0) - \frac{1}{48n^{1/4}}, p^D(z_0) + \frac{1}{48n^{1/4}} \right] \text{ and } C_{z_1^n} = \left[p^D(z_1^n) - \frac{1}{48n^{1/4}}, p^D(z_1^n) + \frac{1}{48n^{1/4}} \right].$$

Note that by the 5th property in Lemma 21, we know that C_{z_0} and $C_{z_1^n}$ are disjoint.

By the 4th property in Lemma 21, we have for any z,

$$r(p^D(z);z) - r(p;z) \geq \frac{2}{3}(p - p^D(z))^2. \tag{B.48}$$

Also by the definition of the regret function, we have

$$\begin{aligned} R_n^\pi(x,T;z_0) &\geq \frac{E_{z_0}^\pi \int_0^1 \{r(p^D(z_0);z_0) - r(p(s);z_0)\}ds}{E_{z_0}^\pi \int_0^1 r(p^D(z_0);z_0)ds} \\ &\geq \frac{4}{3(48)^2\sqrt{n}} E_{z_0}^\pi \int_0^1 I(p(s) \in C_{z_1^n})ds \\ &= \frac{4}{3(48)^2\sqrt{n}} \int_0^1 \mathcal{P}_{z_0}^{\pi(s)}(p(s) \in C_{z_1^n})ds, \end{aligned} \tag{B.49}$$

where the first inequality is because we relaxed the inventory constraint when using π, and the second inequality is because of (B.48), the definition of $C_{z_1^n}$ and that the denominator is 1/2. In the last equality, $\mathcal{P}_{z_0}^{\pi(s)}$ is the probability measure under policy π and up to time s (with underlying demand function has parameter z_0). Similarly, we have

$$\begin{aligned} R_n^\pi(x,T;z_1^n) &\geq \frac{E_{z_1}^\pi \int_0^1 \{r(p^D(z_1^n);z_1^n) - r(p(s);z_1^n)\}ds}{E_{z_1^n}^\pi \int_0^1 r(p^D(z_1^n);z_1^n)ds} \\ &\geq \frac{2}{3(48)^2\sqrt{n}} E_{z_1^n}^\pi \int_0^1 I(p(s) \notin C_{z_1^n})ds \\ &= \frac{2}{3(48)^2\sqrt{n}} \int_0^1 \mathcal{P}_{z_1^n}^{\pi(s)}(p(s) \notin C_{z_1^n})ds, \end{aligned} \tag{B.50}$$

where in the second inequality, we use the fact that the denominator is less than 1, and in the last equality, $\mathcal{P}_{z_1}^{\pi(s)}$ is defined similarly as the one in (B.49).

Now consider any decision rule that maps historical demand observations up to time s into one of the following two sets:

$$H_1 : p(s) \in C_{z_1^n}$$

$$H_2 : p(s) \notin C_{z_1^n}.$$

By Theorem 2.2 in [63], we have the following bound on the probability error of any

decision rule:

$$\mathcal{P}_{z_0}^{\pi(s)}\{p(s) \in C_{z_1^n}\} + \mathcal{P}_{z_1^n}^{\pi(s)}\{p(s) \notin C_{z_1^n}\} \geq \frac{1}{2} e^{-\mathcal{K}(\mathcal{P}_{z_0}^{\pi(s)}, \mathcal{P}_{z_1^n}^{\pi(s)})}. \tag{B.51}$$

However, by the definition of the K-L divergence (3.54), we know that

$$\mathcal{K}(\mathcal{P}_{z_0}^{\pi}, \mathcal{P}_{z_1^n}^{\pi}) - \mathcal{K}(\mathcal{P}_{z_0}^{\pi(s)}, \mathcal{P}_{z_1^n}^{\pi(s)}) \geq E_{z_0}^{\pi} \int_s^1 \frac{1}{2(2 - \frac{1/2 + z_0 - z_0 p(s)}{1/2 + z - z p(s)})} \frac{(z_0 - z)^2 (1 - p(s))^2}{(1/2 + z_0 - z_0 p(s))^2} ds \geq 0, \tag{B.52}$$

where the last inequality is because of (B.45). Therefore, we have

$$\mathcal{P}_{z_0}^{\pi(s)}\{p(s) \in C_{z_1^n}\} + \mathcal{P}_{z_1^n}^{\pi(s)}\{p(s) \notin C_{z_1^n}\} \geq \frac{1}{2} e^{-\mathcal{K}(\mathcal{P}_{z_0}^{\pi}, \mathcal{P}_{z_1^n}^{\pi})}. \tag{B.53}$$

Now we add (B.49) and (B.50) together. We have

$$\begin{aligned} R_n^{\pi}(x, T; z_0) + R_n^{\pi}(x, T; z_1^n) &\geq \frac{2}{3(48)^2\sqrt{n}} \int_0^1 \{\mathcal{P}_{z_0}^{\pi(s)}(p(s) \in C_{z_1^n}) + \mathcal{P}_{z_1^n}^{\pi(s)}(p(s) \notin C_{z_1^n})\} ds \\ &\geq \frac{1}{3(48)^2\sqrt{n}} e^{-\mathcal{K}(\mathcal{P}_{z_0}^{\pi}, \mathcal{P}_{z_1^n}^{\pi})}. \end{aligned}$$

Thus, Lemma 23 holds.

B.5 Proof when the Second-order Derivative Assumption is not Satisfied

Proof of Theorem 6

We first define Algorithm 4 as follows:

Algorithm 4:

Step 1. Initialization

(a) Consider a sequence of $\tau_i, \kappa_i,\ i = 1, 2, ..., N$ (N will be defined later). Define $\underline{p}^1 = \underline{p}$ and $\overline{p}^1 = \overline{p}$. Define $t_i = \sum_{j=1}^{i} \tau_j$, for $i = 0$ to N;

Step 2. Dynamic Learning
For $i = 1$ to N do

(a) Divide $[\underline{p}^i, \overline{p}^i]$ into κ_i equally spaced intervals and let $\{p_{i,j}, j = 1, 2, ..., \kappa_i\}$ be the left endpoints of these intervals;

(b) Divide the time interval $[t_{i-1}, t_i]$ into κ_i parts and define

$$\Delta_i = \frac{\tau_i}{\kappa_i}, \qquad t_{i,j} = t_{i-1} + j\Delta_i \quad j = 0, 1, ..., \kappa_i;$$

(c) Apply $p_{i,j}$ from time $t_{i,j-1}$ to $t_{i,j}$, as long as the inventory is still positive. If no more units are in stock, apply p_∞ until time T and STOP;

(d) Compute

$$\hat{d}(p_{i,j}) = \frac{\text{total demand over } [t_{i,j-1}, t_{i,j}]}{\Delta_i}, \quad j = 1, ..., \kappa_i;$$

(e) Compute

$$\hat{p}_i^u = arg \max_{1 \le j \le \kappa_i} \{p_{i,j}\hat{d}(p_{i,j})\}$$

and

$$\hat{p}_i^c = arg \min_{1 \le j \le \kappa_i} |\hat{d}(p_{i,j}) - x/T|;$$

(f) Set $\hat{p}_i = \max\{\hat{p}_i^c, \hat{p}_i^u\}$. Define

$$\underline{p}^{i+1} = \hat{p}_i - \frac{\log n}{2} \cdot \frac{\overline{p}^i - \underline{p}^i}{\kappa_i}$$

and

$$\overline{p}^{i+1} = \hat{p}_i + \frac{\log n}{2} \cdot \frac{\overline{p}^i - \underline{p}^i}{\kappa_i}.$$

And the price range for the next iteration

$$I_{i+1} = [\underline{p}^{i+1}, \overline{p}^{i+1}].$$

Here we truncate the interval if it doesn't lie inside the feasible set of $[\underline{p}, \overline{p}]$;

Step 3. Applying the optimal price

(a) Use $\hat{p}_N$ for the rest of time until the stock is run out.

Similarly as we study Algorithm 3, we first list the set of equations we want the parameters to satisfy:

$$\frac{\overline{p}_i - \underline{p}_i}{\kappa_i} \sim \sqrt{\frac{\kappa_i}{n\tau_i}}, \quad \forall i = 1, ..., N, \tag{B.54}$$

$$\overline{p}_{i+1} - \underline{p}_{i+1} \sim \log n \cdot \frac{\overline{p}_i - \underline{p}_i}{\kappa_i}, \quad \forall i = 1, ..., N-1, \tag{B.55}$$

$$\tau_{i+1} \cdot \frac{\overline{p}_i - \underline{p}_i}{\kappa_i} \cdot \sqrt{\log n} \sim \tau_1, \quad \forall i = 1, ..., N-1. \tag{B.56}$$

Also we define

$$N = \min_l \{l | \frac{\sqrt{\log n} \cdot (\overline{p}_l - \underline{p}_l)}{\kappa_l} < \tau_1\}. \tag{B.57}$$

The meaning of each of these equations is similar to the one in Algorithm 3 (but since we have a different local behavior of the revenue function under this alternative assumption, we have different relationships between these parameters). We solve κ_i and τ_i from the above relations. Define $\tau_1 = n^{-\frac{1}{2}} \cdot (\log n)^3$. We get

$$\kappa_i = n^{\frac{1}{6} \cdot (\frac{2}{3})^{i-1}} \log n, \quad \forall i, \tag{B.58}$$

$$\tau_i = n^{-\frac{1}{2}(\frac{2}{3})^{i-1}} (\log n)^3, \quad \forall i. \tag{B.59}$$

And as a by-product, we have

$$\overline{p}_i - \underline{p}_i = n^{-\frac{1}{2}(1-(\frac{2}{3})^{i-1})}, \quad \forall i. \tag{B.60}$$

Now we prove some lemmas similar to those we used to prove Theorem 4.

Lemma 27 *N defined in (B.57) satisfy: $N \leq \log n$.*

Proof. We plug (B.58) and (B.60) into (B.57) and the result follows. □

Now we prove a lemma similar to Lemma 12, showing that the price range for each learning period contains the actual optimal price, with high probability.

Lemma 28 *Denote the optimal deterministic price by p^D. And the assumption that $L|p^u - p| \leq |r(\lambda(p^u)) - r(\lambda(p))|$ hold. Then with probability $1 - O\left(\frac{1}{n}\right)$, $p^D \in I_i$ for any $i = 1, ..., N$.*

Proof. Like the proof of Lemma 12, we first show that if $p^D \in I_i$, then with probability $1 - O\left(\frac{1}{n^2}\right)$, $p^D \in I_{i+1}$. Define

$$u_n^i = 2\sqrt{\log n} \max\left\{\frac{\overline{p}_i - \underline{p}_i}{\kappa_i}, \sqrt{\frac{\kappa_i}{n\tau_i}}\right\}. \tag{B.61}$$

We consider three cases:

- $p_i^u < p^u$: This is impossible since we know that $p^D \geq p^u > p_i^u$ and by the induction assumption $p^D \in I_i$. Therefore, we must have $p^u \in I_i$ and by definition, p^u achieves larger revenue rate than p_i^u, which is contradictory to the definition of p_i^u

- $p_i^u = p^u$: We assume p_{i,j^*} is the nearest grid point to p_i^u in this iteration. In this case, by the granularity of the grid at iteration i, we have $|p_{i,j^*} - p^u| \leq \frac{\overline{p}_i - \underline{p}_i}{\kappa_i}$ and thus by our assumption that $r(p)$ is Lipschitz continuous, we know that $|r(\lambda(p^u)) - r(\lambda(p_{i,j^*}))| \leq C \cdot (\frac{\overline{p}_i - \underline{p}_i}{\kappa_i})$, therefore we have:

$$\begin{aligned} & r(\lambda(p^u)) - r(\lambda(\hat{p}_i^u)) \\ = \; & r(\lambda(p^u)) - r(\lambda(p_{i,j^*})) + p_{i,j}^u \lambda(p_{i,j^*}) - p_{i,j^*}^u \hat{\lambda}(p_{i,j^*}) \\ & -(\hat{p}_i^u \lambda(\hat{p}_i^u) - \hat{p}_i^u \hat{\lambda}(\hat{p}_i^u)) + p_{i,j^*} \hat{\lambda}(p_{i,j^*}) - \hat{p}_i^u \hat{\lambda}(\hat{p}_i^u) \\ \leq \; & C(\tfrac{\overline{p}_i - \underline{p}_i}{\kappa_i}) + 2 \max_{1 \leq j \leq \kappa_i} |p_{i,j} \lambda(p_{i,j}) - p_{i,j} \hat{\lambda}(p_{i,j})|. \end{aligned} \tag{B.62}$$

In (B.62), $\hat{\lambda}$ is the observed demand rate, and the last inequality is due to the definition of $\hat{p}_i^u$ and that $\hat{p}_i^u$ is among one of the $p_{i,j}$.

By Lemma 26 in Appendix B.2, we have

$$P(|\hat{\lambda}(p_{i,j}) - \lambda(p_{i,j})| > C\sqrt{\log n} \cdot \sqrt{\frac{\kappa_i}{n\tau_i}}) \leq \frac{1}{n^2} \tag{B.63}$$

with some suitable constant C. Therefore, with probability $1-O(\frac{1}{n^2})$, $r(\lambda(p^u))-r(\lambda(\hat{p}_i^u)) \leq Cu_n^i$. However, by our assumption that $L|p^u - p| \leq |r(\lambda(p^u)) - r(\lambda(p))|$, with probability $1 - O\left(\frac{1}{n}\right)$, $|p^u - \hat{p}_i^u| \leq Cu_n^i$.

Now we consider the distance between $\hat{p}_i^c$ and p_i^c. Assume p_{i,j^*} is the nearest grid point to p_i^c. Then we have

$$\begin{aligned} |\lambda(\hat{p}_i^c) - x| &\leq |\hat{\lambda}(\hat{p}_i^c) - x| + |\hat{\lambda}(\hat{p}_i^c) - \lambda(\hat{p}_i^c)| \\ &\leq |\hat{\lambda}(p_{i,j^*}) - x| + |\hat{\lambda}(\hat{p}_i^c) - \lambda(\hat{p}_i^c)| \\ &\leq |\lambda(p_{i,j^*}) - x| + |\hat{\lambda}(p_{i,j^*}) - \lambda(p_{i,j^*})| + |\hat{\lambda}(\hat{p}_i^c) - \lambda(\hat{p}_i^c)| \\ &\leq |\lambda(p_i^c) - x| + |\lambda(p_i^c) - \lambda(p_{i,j^*})| + 2\max_{1\leq j\leq\kappa_i} |\hat{\lambda}(p_{i,j}) - \lambda(p_{i,j})|. \end{aligned} \tag{B.64}$$

And by the definition of p_i^c, $\lambda(p_i^c)-x$ and $\lambda(\hat{p}_i^c)-x$ must have the same sign, otherwise there exists a point in between that achieves smaller $|\lambda(p)-x|$. Therefore we have

$$|\lambda(p_i^c) - \lambda(\hat{p}_i^c)| \leq |\lambda(p_i^c) - \lambda(p_{i,j^*})| + 2\max_{1\leq j\leq\kappa_i} |\hat{\lambda}(p_{i,j}) - \lambda(p_{i,j})|. \tag{B.65}$$

By the Lipschitz continuity of λ, we have

$$|\lambda(p_i^c) - \lambda(p_{i,j^*})| \leq \frac{\overline{p}_i - \underline{p}_i}{\kappa_i}. \tag{B.66}$$

Also by Lemma 26 in Appendix B.2, we have with probability $1 - O\left(\frac{1}{n^2}\right)$,

$$\max_{1\leq j\leq\kappa_i} |\hat{\lambda}(p_{i,j}) - \lambda(p_{i,j})| \leq C\sqrt{\log n} \cdot \sqrt{\frac{\kappa_i}{n\tau_i}}. \tag{B.67}$$

Therefore, with probability $1 - O\left(\frac{1}{n^2}\right)$, we have

$$|\lambda(p_i^c) - \lambda(\hat{p}_i^c)| \leq C u_n^i, \tag{B.68}$$

and by the Lipschitz continuity of $\nu(\lambda)$, this implies that with probability $1 - O\left(\frac{1}{n^2}\right)$,

$$|\hat{p}_i^c - p_i^c| \leq C u_n^i. \tag{B.69}$$

Therefore, we have

$$\begin{aligned} & P\{|\hat{p}_i - p^D| > C u_n^i\} \\ \leq \; & P\{|\hat{p}_i^c - p_i^c| > C u_n^i\} + P\{|\hat{p}_i^u - p^u| > C u_n^i\} \leq O\left(\frac{1}{n^2}\right). \end{aligned} \tag{B.70}$$

Here we used the fact that:

$$|\max\{a, c\} - \max\{b, d\}| > u \quad \Rightarrow |a - b| > u \text{ or } |c - d| > u.$$

Note that (B.70) is equivalent of saying that

$$P(p^D \in [\hat{p}_i - C u_n^i, \hat{p}_i + C u_n^i]) > 1 - O\left(\frac{1}{n}\right). \tag{B.71}$$

Now also note that the interval I_{i+1} in our algorithm is chosen to be

$$\left[\hat{p}_i - \frac{\log n}{2} \frac{\overline{p}_i - \underline{p}_i}{\kappa_i}, \hat{p}_i + \frac{\log n}{2} \frac{\overline{p}_i - \underline{p}_i}{\kappa_i}\right],$$

which is of order $\sqrt{\log n}$ greater than $\sqrt{u_n^i}$ (and according to the way we defined κ_i and τ_i, the two terms in u_n^i are of the same order). Therefore we know that with probability $1 - O(\frac{1}{n^2})$, $p^D \in I_{i+1}$.

- $p^u < p_i^u$: In this case, $p^D = p^c$. With the same argument, but only the p^c part, we know that with probability $1 - O\left(\frac{1}{n^2}\right)$, $p^D \in I_{i+1}$.

Also, as claimed in the previous lemma, the number of steps N is less than $\log n$. Therefore, we can take a union bound over N steps, and claim that with probability $1 - O\left(\frac{1}{n}\right)$, $p^D \in I_i$, for all $i = 1, ..., N$. $\square$

Now we have proved that with high probability, p^D will always be in our interval. Next we will analyze the revenue collected by this mechanism.

Define Y_{ij} to be the Poisson random variable with parameter $\lambda(p_{i,j})n\Delta_i$ ($Y_{ij} = N(\lambda(p_{i,j})n\Delta_i)$. Also define $\hat{Y}$ to be a Poisson random variable with parameter $\lambda(\hat{p}_N)n(1-t_N)$ ($\hat{Y} = N(\lambda(\hat{p}_N)n(1-t_N))$. We define the following event:

$$A_1 = \{\omega : \sum_{i,j} Y_{ij} < nx\}.$$

We have

$$J_n^\pi(x, T; \lambda) \geq E[\sum_{i=1}^{N}\sum_{j=1}^{\kappa_i} p_{i,j}Y_{ij}I(A_1)] + E[\tilde{p}\min(\hat{Y}, (nx - \sum_{i,j} Y_{ij})^+)I(A_1)]. \quad \text{(B.72)}$$

In the following, we will consider each term in (B.72). We will show that the revenue collected in both parts is "close" to the revenue generated by the optimal deterministic price p^D in its corresponding part (and the consumed inventory is also near-optimal). We first have:

Lemma 29

$$E[\sum_{i=1}^{N}\sum_{j=1}^{\kappa_i} p_{i,j}Y_{ij}I(A_1)] \geq \sum_{i=1}^{N} p^D\lambda(p^D)n\tau_i - Cn\tau_1\log n. \quad \text{(B.73)}$$

Proof. The proof of this lemma is almost the same to the proof of Lemma 14 in Appendix B.3, except that in (B.21), $m_L K(\overline{p}_i - \underline{p}_i)^2$ is replaced by $L(\overline{p}_i - \underline{p}_i)$. In (B.22), We use the relationship defined in (B.56), the result follows. $\square$

Now we look at the other term in (B.72). We have

$$\begin{aligned} E[\hat{p}_N \min(\hat{Y}, (nx - \sum_{i,j} Y_{ij})^+)] &= E[\hat{p}_N(\hat{Y} - \max(\hat{Y} - (nx - \sum_{i,j} Y_{ij})^+, 0))] \\ &\geq E[\hat{p}_N \hat{Y}] - E[\hat{p}_N(\hat{Y} + \sum_{i,j} Y_{ij} - nx)^+]. \end{aligned}$$

For $E[\hat{p}_N \hat{Y}] = E[\hat{p}_N \lambda(\hat{p}_N) n(1 - t_N)]$, we apply the same argument as we proved in Lemma 29, and we have:

$$E[r(\lambda(\hat{p}_N))] \geq r(\lambda(p^D)) - C u_N, \tag{B.74}$$

where $u_N = 2\sqrt{\log n} \cdot \max\{\frac{\overline{p}_N - \underline{p}_N}{\kappa_N}, \sqrt{\frac{\kappa_N}{n\tau_N}}\}$. Therefore,

$$E[\hat{p}_N \hat{Y}] \geq p^D \lambda(p^D) \cdot n(1 - t_n) - Cnu_N \geq p^D \lambda(p^D) \cdot n(1 - t_n) - Cn\tau_1 \log n. \tag{B.75}$$

Now we consider

$$E[\hat{p}_N(\hat{Y} + \sum_{i,j} Y_{ij} - nx)^+]. \tag{B.76}$$

First we relax this to

$$\overline{p} E(\hat{Y} + \sum_{i,j} Y_{ij} - nx)^+.$$

We claim that:

Lemma 30

$$E(\hat{Y} + \sum_{i,j} Y_{ij} - E\hat{Y} - \sum_{i,j} EY_{ij})^+ \leq Cn\tau_1 \log n, \tag{B.77}$$

where C is a properly chosen constant.

and

Lemma 31

$$\sum_{i,j|} EY_{ij} + E\hat{Y} - nx \leq Cn\tau_1 \log n, \tag{B.78}$$

where C is a properly chosen constant.

The proof of Lemma 30 is exact the same as the proof of Lemma 15. For Lemma 31, we have:

$$EY_{ij} = \lambda(p_{i,j})n\Delta_i, \tag{B.79}$$

$$E\hat{Y} = \lambda(\hat{p}_N)n(1-t_n). \tag{B.80}$$

By our assumption that p^D is in the interior and Lemma 28, we know that with probability $1 - O\left(\frac{1}{n}\right)$, $p^c < \overline{p}_i$ for all our price ranges. Therefore, with probability $1 - O\left(\frac{1}{n}\right)$, we have

$$p^c - p_{i,j} \leq \overline{p}_i - \underline{p}_i. \tag{B.81}$$

By the Lipschitz condition on $\lambda(p)$, this implies that $\lambda(p_{i,j}) - \lambda(p^c) \leq C(\overline{p}_i - \underline{p}_i)$. Therefore, with probability $1 - O\left(\frac{1}{n}\right)$,

$$\begin{aligned} \textstyle\sum_{i,j} EY_{ij} - nxt_N &= \textstyle\sum_{i,j}(EY_{ij} - \lambda(p^c)n\Delta_i) \\ &\leq Cn\textstyle\sum_{i,j}(\overline{p}_i - \underline{p}_i)\Delta_i \\ &= Cn\tau_1 \log n. \end{aligned} \tag{B.82}$$

Similarly, we have that with probability $1 - O\left(\frac{1}{n}\right)$,

$$\lambda(\hat{p}_N) - \lambda(p^c) \leq \frac{\overline{p}_N - \underline{p}_N}{\kappa_N} \leq C\tau_1.$$

And therefore,

$$E\hat{Y} - nx(1-t_N) = E\hat{Y} - \lambda(p^c)n(1-t_n) \leq Cn\tau_1. \tag{B.83}$$

Thus the lemma holds. □

We combine Lemma 29, 30 and 31, Theorem 6 holds.

Bibliography

[1] D. Adelman. Dynamic bid prices in revenue management. *Operations Research*, 55(4):647–661, 2007.

[2] R. Agrawal. The continuum-armed bandit problem. *SIAM Journal of Control and Optimization*, 33(6):1926–1951, 1995.

[3] S. Agrawal. Online multi-item multi-unit auctions. Technical report, Stanford University, 2008.

[4] S. Agrawal, Z. Wang, and Y. Ye. A dynamic near-optimal algorithm for online linear programming. Submitted, 2010.

[5] V. F. Araman and R. A. Caldentey. Dynamic pricing for nonperishable products with demand learning. *Operations Research*, 57(5):1169–1188, 2009.

[6] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire. The non-stochastic multiarmed bandit problem. *SIAM Journal on Computing*, 32(1):48–77, 2002.

[7] Y. Aviv and A. Pazgal. Pricing of short life-cycle products through active learning. Technical report, Washington University, 2002.

[8] B. Awerbuch, Y. Azar, and S. Plotkin. Throughput-competitive on-line routing. In *FOCS'93: Proceeding of the 34th Annual Symposium on Foundations of Computer Science*, pages 32–40, 1993.

[9] M. Babaioff, N. Immorlica, D. Kempe, and R. Kleinberg. Online auctions and generalized secretary problems. *SIGecom Exch.*, 7(2):1–11, 2008.

[10] M. Babaioff, N. Immorlica, and R. Kleinberg. Matroids, secretary problems, and online mechanisms. In *SODA'07: Proceedings of the 18th annual ACM-SIAM symposium on Discrete algorithms*, pages 434–443, 2007.

[11] M-F. Balcan, A. Blum, J. D. Hartline, and Y. Mansour. Mechanism design via machine learning. In *FOCS'05: Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*, pages 605–614, 2005.

[12] M. Ball and M. Queyranne. Towards robust revenue management competitive analysis of online booking. *Operations Research*, 57(4):950–963, 2009.

[13] D. Bertsimas and G. Perakis. Dynamic pricing: A learning approach. In *Mathematical and Computational Models for Congestion Charging*, volume 101 of *Applied Optimization*, pages 45–79. Springer US, 2006.

[14] O. Besbes and A. Zeevi. Dynamic pricing without knowing the demand function: Risk bounds and near-optimal algorithms. *Operations Research*, 57(6):1407–1420, 2009.

[15] O. Besbes and A. Zeevi. Blind network revenue management. Submitted, 2010.

[16] O. Besbes and A. Zeevi. On the (surprising) sufficiency of linear models for dynamic pricing with demand learning. Submitted, 2012.

[17] G. Bitran and R. Caldentey. An overview of pricing models for revenue management. *Manufacturing and Service Operations Management*, 5(3):203–229, 2003.

[18] P. Bremaud. *Point Process and Queues: Martingale Dynamics*. Springer-Verlag, 1980.

[19] J. Broder and P. Rusmevichientong. Dynamic pricing under a general parametric choice model. *Operations Research*, 2012. forthcoming.

[20] N. Buchbinder, K. Jain, and J. Naor. Online primal-dual algorithms for maximizing ad-auctions revenue. In *Algorithms–ESA 2007*, pages 253–264, 2007.

[21] N. Buchbinder and J. Naor. Improved bounds for online routing and packing via a primal-dual approach. In *FOCS'06: Proceedings of the 47th Annual IEEE Symposium on Foundations of Computer Science*, pages 293–304, 2006.

[22] N. Buchbinder and J. Naor. Online primal-dual algorithms for covering and packing. *Mathematics of Operations Research*, 34(2):270–286, 2009.

[23] A. Carvalho and M. Puterman. Learning and pricing in an internet environment with binomial demands. *Journal of Revenue and Pricing Management*, 3(4):320–336, 2005.

[24] S. Chawla, J. D. Hartline, D. Malec, and B. Sivan. Sequential posted pricing and multi-parameter mechanism design. In *STOC'10: Proceedings of the 42nd ACM symposium on Theory of computing*, pages 311–320, 2010.

[25] W. L. Cooper. Asymptotic behavior of an allocation policy for revenue management. *Operations Research*, 50(4):720–727, 2002.

[26] W. L. Cooper, T. Homem de Mello, and A. Kleywegt. Models of the spiral-down effect in revenue management. *Operations Research*, 54(5):968–987, 2006.

[27] W. L. Cooper, T. Homem de Mello, and A. Kleywegt. Learning and pricing with models that do not explicitly incorporate competition. Submitted, 2012.

[28] R. Cross. *Revenue Management: Hard-Core Tactics for Market Domination*. Broadway Books, 1997.

[29] G. Dantzig. *Linear Programming and Extensions*. Princeton University Press, August 1963.

[30] A. den Boer and B. Zwart. Simultaneously learning and optimizing using controlled variance pricing. Submitted, 2011.

[31] N. R. Devanur. Online algorithms with stochastic input. *SIGecom Exchange*, 10(2):40–49, 2011.

[32] N. R. Devanur and T. P. Hayes. The adwords problem: online keyword matching with budgeted bidders under random permutations. In *EC'09: Proceedings of the tenth ACM conference on Electronic Commerce*, pages 71–78, 2009.

[33] N. R. Devanur, K. Jain, B. Sivan, and C. A. Wilkens. Near optimal online algorithms and fast approximation algorithms for resource allocation problems. In *EC'11: Proceedings of the twelfth ACM conference on Electronic Commerce*, pages 29–38, 2011.

[34] W. Elmaghraby and P. Keskinocak. Dynamic pricing in the presence of inventory considerations: Research overview, current practices and future directions. *Management Science*, 49(10):1287–1389, 2003.

[35] V. F. Farias and B. Van Roy. Dynamic pricing with a prior on market response. *Operations Research*, 58(1):16–29, 2010.

[36] J. Feldman, M. Henzinger, N. Korula, V. Mirrokni, and C. Stein. Online stochastic packing applied to display ad allocation. *Algorithms–ESA 2010*, pages 182–194, 2010.

[37] J. Feldman, N. Korula, V. Mirrokni, S. Muthukrishnan, and M. Pal. Online ad assignment with free disposal. In *WINE'09: Proceedings of the fifth Workshop on Internet and Network Economics*, pages 374–385, 2009.

[38] J. Feldman, A. Mehta, V. Mirrokni, and S. Muthukrishnan. Online stochastic matching: Beating 1 - 1/e. In *FOCS'09: Proceedings of the 50th Annual IEEE Symposium on Foundations of Computer Science*, pages 117–126, 2009.

[39] G. Gallego and G. van Ryzin. Optimal dynamic pricing of inventories with stochastic demand over finite horizons. *Management Science*, 40(8):999–1020, 1994.

[40] G. Gallego and G. van Ryzin. A multiproduct dynamic pricing problem and its application to network yield management. *Operations Research*, 45(1):24–41, 1997.

[41] G. Goel and A. Mehta. Online budgeted matching in random input models with applications to adwords. In *SODA'08: Proceedings of the nineteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 982–991, 2008.

[42] M. Harrison, B. Keskin, and A. Zeevi. Bayesian dynamic pricing policies: Learning and earning under a binary prior distribution. *Management Science*, 2011. forthcoming.

[43] M. Harrison, B. Keskin, and A. Zeevi. Dynamic pricing with an unknown linear demand model: Asymptotically optimal semi-myopic policies. Submitted, 2011.

[44] S. Jasin and S. Kumar. A re-solving heuristic with bounded revenue loss for network revenue management with customer choice. *Mathematics of Operations Research*, 2012.

[45] R. Kleinberg and F. T. Leighton. The value of knowing ad demand curve: Bounds on regret for online posted-price auctions. In *FOCS'05: Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*, pages 594–605, 2005.

[46] T. L. Lai and H. Robbins. Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics*, 6(1):4–22, 1985.

[47] A. Lim and J. Shanthikumar. Relative entropy, exponential utility, and robust dynamic pricing. *Operations Research*, 55(2):198–214, 2007.

[48] G. Y. Lin, Y. Lu, and D. Yao. The stochastic knapsack revisited: Switch-over policies and dynamic pricing. *Operations Research*, 56(4):945–957, 2008.

[49] M. S. Lobo and S. Boyd. Pricing and learning with uncertain demand. Technical report, Duke University, 2003.

[50] C. Maglaras and J. Meissner. Dynamic pricing strategies for multiproduct revenue management problems. *Manufacturing and Service Operations Management*, 8(2):136–148, 2006.

[51] M. Mahdian and Q. Yan. Online bipartite matching with random arrivals: an approach based on strongly factor-revealing lps. In *STOC'11: Proceedings of the the 43rd annual ACM symposium on Theory of Computing*, 2011.

[52] J. McGill and G. van Ryzin. Revenue management: Research overviews and prospects. *Transportation Sciences*, 32(2):233–256, 1999.

[53] A. Mehta, A. Saberi, U. Vazirani, and V. Vazirani. Adwords and generalized online matching. In *FOCS'05: Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*, pages 264–273, 2005.

[54] P. Orlik and H. Terao. *Arrangement of Hyperplanes*. Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences], Springer-Verlag, Berlin, 1992.

[55] G. Perakis and G. Roels. The "price of information": Inventory management with limited information about demand. *Manufacturing and Service Operations Management*, 8(1):98–117, 2006.

[56] R. Phillips. *Pricing and Revenue Optimization*. Stanford University Press.

[57] P. Rusmevichientong, B. Van Roy, and P. Glynn. A non-parametric approach to multi-product pricing. *Operations Research*, 54(1):82–98, 2010.

[58] R. W. Simpson. Using network flow techniques to find shadow prices for market and seat inventory control. *MIT Flight Transportation Laboratory Memorandum M89-1, Cambridge, MA*, 1989.

[59] B. C. Smith, J. F. Leimkuhler, and R. M. Darrow. Yield management at american airlines. *Interfaces*, 22:8–31, 1992.

[60] K. Talluri and G. van Ryzin. An analysis of bid-price controls for network revenue management. *Management Science*, 44(11):1577–1593, 1998.

[61] K. T. Talluri and G. J. van Ryzin. *Theory and Practice of Revenue Management*. Springer-Verlag, 2005.

[62] H. Topaloglu. Using lagrangian relaxation to compute capacity-dependent bid prices in network revenue management. *Operations Research*, 57(3):637–649, 2009.

[63] A. Tsybakov. *Introduction to Nonparametric Estimation*. Springer-Verlag, 2004.

[64] A. van der Vaart and J. Wellner. *Weak Convergence and Empirical Processes: With Applications to Statistics (Springer Series in Statistics)*. Springer, November 1996.

[65] G. van Ryzin and J. McGill. Revenue management without forecasting or optimization: An adaptive algorithm for determining airline seat protection levels. *Management Science*, 46(6):760–775, 2000.

[66] V. Vazirani, N. Nisan, T. Roughgarden, and E. Tardos. *Algorithmic Game Theory*. Cambridge University Press, 2007.

[67] Z. Wang, S. Deng, and Y. Ye. Close the gaps: A learning-while-doing algorithm for a class of single-product revenue management problems. Submitted, 2011.

[68] E. L. Williamson. Airline network seat control. *Ph. D. Thesis, MIT, Cambridge, MA*, 1992.

[69] W. Zhao and Y. Zheng. Optimal dynamic pricing for perishable assets with nonhomogeneous demand. *Management Science*, 46(3):375–388, 2000.