Data-Driven Pricing & Optimization

van de Geer, R.

2019

document version
Publisher's PDF, also known as Version of record

Data-Driven Pricing & Optimization

Dissertation Committee

chairman: prof.dr. V. Subramaniam
Vrije Universiteit Amsterdam, Amsterdam, the Netherlands

promotor: prof.dr. Sandjai Bhulai
Vrije Universiteit Amsterdam, Amsterdam, the Netherlands
copromotor: dr. Arnoud V. den Boer
Universiteit van Amsterdam, Amsterdam, the Netherlands

committee: prof.dr. Ger Koole
Vrije Universiteit Amsterdam, Amsterdam, the Netherlands
prof.dr. Rob van der Mei
Vrije Universiteit Amsterdam, Amsterdam, the Netherlands
dr. Flora Spieksma
Leiden University, Leiden, the Netherlands
dr. George Chen
London Business School, London, United Kingdom
dr. Alwin Haensel
Haensel AMS, Berlin, Germany

ISBN/EAN: 978-94-028-1474-3

Typeset by LaTeX.
Printed by: Ipskamp Printing
Cover design by: Solange Beeks, Amsterdam, The Netherlands

VRIJE UNIVERSITEIT

Data-Driven Pricing & Optimization

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad Doctor aan
de Vrije Universiteit Amsterdam,
op gezag van de rector magnificus
prof.dr. V. Subramaniam,
in het openbaar te verdedigen
ten overstaan van de promotiecommissie
van de Faculteit der Bètawetenschappen
op dinsdag 4 juni om 9.45 uur
in de aula van de universiteit,
De Boelelaan 1105

door

Ruben van de Geer

geboren te Koog aan de Zaan

promotor: prof.dr. S. Bhulai
copromotor: dr. A.V. den Boer

I blame all of you. Writing this book has been an exercise in sustained suffering. The casual reader may, perhaps, exempt herself from excessive guilt, but for those of you who have played the larger role in prolonging my agonies with your encouragement and support, well... you know who you are, and you owe me.

Brendan Pietsch (2015)

Preface

Although it is only my name that graces the cover of this thesis, you should know that many people contributed to it, in one way or another. Today, finally, is the time to express my gratitude to those people.

First and foremost, I would like to thank my supervisor Sandjai Bhulai. I am eternally grateful for the positive environment with which you provided me. The research presented in this work would not have been possible without your continuous enthusiasm, faith, and encouragement. Also, I would like to express my deepest gratitude to Stefan van Duin (Deloitte Consulting) and Frans Feldberg (Vrije Universiteit Amsterdam) for making the research project that led to this dissertation possible. Thank you both, for all your encouragement and for starting this endeavor in the first place.

I am very thankful to Arnoud den Boer (co-author and co-promotor) and Qingchen Wang (co-author) for a joyful and fruitful collaboration. Arnoud, the process of writing the paper that led to Chapter 2 required all our persistence and dedication. We kept the spirit alive at all times—an accomplishment in itself—and, in the end, we were able to deliver. Thank you for teaching me countless invaluable lessons. Qingchen, together we wrote the paper that led to Chapter 5. Thank you for introducing me to the field of machine learning, for which you seem to have developed a sixth sense. It was great working with you.

I have fond memories from working at the Analytics & Optimization (A&O) group at the Vrije Universiteit and at the Analytics and Information Management group at Deloitte Consulting. In particular, I thank Ger Koole of A&O for organizing outings and events on numerous occasions, as well as for luring me into joining him for a full-distance triathlon in 2017. Furthermore, I thank my fellow researchers for their good company—in particular the other full-time inhabitants of the R5 wing, namely Siqiao "April" Li, Jaap Storm, and Marijn ten Thij.

All that remains, is for me to express my deepest gratitude to my family, without whom

I would never have gotten to the point of defending my dissertation. Mum and dad, thank you for your unconditional love and support. Annemarie and Jorrit, thank you for having my back during the defense of this dissertation. It is an honor to have you sitting next to me. Solange, I am eternally grateful for your presence in my life and relish every moment spent with you.

Ruben van de Geer
March 2019, Amsterdam

Contents

Chapter 1

Introduction and Contributions

This dissertation covers various topics on data-driven pricing and optimization. Each chapter treats a specific topic and is self-contained, which allows the reader to consume any subset of chapters in any particular order. The first three chapters revolve around the problem of setting the *right* price for a product or an assortment of multiple products. We consider this problem from a theoretical (Ch. 2), numerical (Ch. 3), and empirical (Ch. 4) point of view. The final chapter (Ch. 5) considers data-driven optimization in the context of a scheduling problem. Below follows a detailed overview of the content and contributions of each chapter.

Chapter 2: Price Optimization under the Finite-Mixture Logit Model

In Chapter 2, we consider the problem of optimizing prices (with respect to revenue) for a monopolist that sells products to customers that choose according to the finite-mixture logit (FML) model. This chapter fits the classical approach—pioneered by the French philosopher and mathematician Antoine Augustin Cournot in 1838—in which first the relationship between price and demand is modeled as a mathematical function and, subsequently, the price is set that optimizes a certain objective function (e.g., the revenue or profit).

The FML choice model assumes customers belong to one of a number of customer segments, where each customer segment chooses according to a multinomial logit model with segment-specific parameters. As such, the FML choice model incorporates customer heterogeneity, i.e., it acknowledges that different *types* of customers exist that have different preferences. Existing work in the field of price optimization has consid-

ered customer choice models—most notably the multinomial logit and the nested logit model—that assume customers are homogeneous. By considering the pricing problem under FML, we depart from this stringent assumption.

Our approach is to develop a novel characterization of the price optimization problem under FML. Leveraging this characterization, we construct an algorithm that obtains prices at which the revenue is guaranteed to be at least $(1 - \epsilon)$ times the maximum attainable revenue for any prespecified $\epsilon > 0$. Existing global optimization methods require exponential time in the number of products to obtain such a result. Practically, this means that only the prices of a handful of products can be optimized. The running time of our algorithm, however, is exponential in the number of customer segments and only polynomial in the number of products. This is of great practical value, since in applications the number of products can be very large, while it is has been found in various contexts that a low number of segments is sufficient to capture customer heterogeneity appropriately. The results of our numerical study show that our algorithm runs fast on a broad range of problem instances. As such, our work improves the applicability of price optimization in business practice, where the importance of accounting for customer heterogeneity has long been acknowledged.

Chapter 2 is based on van de Geer and den Boer (2018).

Chapter 3: Numerical Performance of Dynamic Pricing and Learning Algorithms: A Controlled Experiment

Chapter 3 considers the problem of dynamic pricing and learning in a competitive environment. The *learning* here pertains to the fact that the seller does not know in advance how price affects demand (e.g., because the seller does not have any historical data at its disposal to model this relationship). By experimenting with prices, the seller can *learn* the price-demand relationship and optimize prices accordingly.

Evidently, the question remains what a good approach (or algorithm) is to "learn and earn". The case of pricing and learning in the presence of competition has proven to be a very challenging problem for the simple reason that the performance of an algorithm depends on the (unknown) pricing behavior of competitors. A particular strategy may work very well when used against simple pricing rules, but perform much worse against sophisticated algorithms. Not only is theoretical understanding limited; there also does not appear to be an extensive numerical study that compares the practical performance of different algorithms.

This motivated us to organize the Dynamic Pricing Challenge, held on the occasion of the 17th INFORMS Revenue Management and Pricing Section Conference on June 29-30, 2017, at the Centrum Wiskunde & Informatica, Amsterdam, The Netherlands. For this challenge, participants were invited to submit dynamic pricing and learning algorithms that were subsequently used to simulate a broad range of market environments in both duopoly and oligopoly settings. The extensive simulations that we ran allow us to describe the numerical performance of various pricing and learning algorithms and provides insight into the performance and properties of several types of policies. Given that the participants submitted a wide variety of algorithms—such as bandit-type models, customer choice models, econometric regression models, machine learning models, and greedy ad-hoc approaches—we are able to relate the performance of a broad range of algorithms to different market structures.

Chapter 3 is based on van de Geer et al. (2018a).

Chapter 4: Bayesian Modeling of Customer Choice Across Product Categories for Promotion Optimization

This chapter entails an empirical study on modeling customer choice behavior at an apparel retailer with an application to optimization of personalized promotions. Using transaction data of a leading department store, we infer price sensitivities and brand preferences on a customer level across various product categories by relying on Bayesian customer choice modeling. More precisely, we assume that customers are utility maximizers and measure how brand and price affect customer utility within categories, as well as the dependency in price sensitivity and brand preferences across categories. As such, this study is the first in a stream of research on multi-category modeling that considers durable goods, whereas existing studies focus on grocery retailing.

We find that customers exhibit similar purchasing behavior across categories. To exploit this, our framework allows us to map a customer's preferences in one category onto categories in which this customer did not purchase before. To show the practical relevance in targeted marketing of this, we present an algorithm that leverages the customer-specific estimates to optimize personalized promotions, including promotions in categories in which customers did not purchase before.

Chapter 4 is based on van de Geer and Bhulai (2018).

Chapter 5: Data-driven Consumer Debt Collection via Machine Learning and Approximate Dynamic Programming

Chapter 5 develops and tests a framework for the data-driven scheduling of outbound calls made by debt collectors. These phone calls are used to persuade debtors to settle their debt, or to negotiate payment arrangements in case debtors are willing, but unable to repay. We determine on a daily basis which debtors should be called to maximize the amount of delinquent debt recovered in the long term, under the constraint that only a limited number of phone calls can be made each day.

Our approach is to formulate a Markov decision process and, given its intractability, approximate the value function based on historical data through the use of state-of-the-art machine learning techniques. Specifically, we predict the likelihood with which a debtor in a particular state is going to settle its debt and use this as a proxy for the value function. Based on this value function approximation, we compute for each debtor the *marginal value* of making a call. This leads to a particularly straightforward optimization procedure, namely, we prioritize the debtors that have the highest marginal value per phone call.

We validate our proposed methodology in a controlled field experiment conducted with real debtors. The results show that our optimized policy substantially outperforms the current scheduling policy that has been used in business practice for many years. Most importantly, our policy collects more debt in less time, whilst using substantially fewer resources—leading to a large increase in the amount of debt collected per phone call.

Chapter 5 is based on van de Geer et al. (2018b).

Scientific publications not contained in this thesis

- Moeke, D., van de Geer, R., Koole, G., and Bekker, R. (2016). On the performance of small-scale living facilities in nursing homes: A simulation approach. *Operations research for health care*, 11:20–34.

Publications in popular media for the valorization of this research

- *Data-Driven Debt Collection Using Machine Learning and Predictive Analytics*
  Blog post on DataScience.com (July 26, 2018) available at:

https://www.datascience.com/blog/data-driven-debt-collection-machine-learning-predictive-analytics

– *Als het kouder wordt, zakt ook de prijs van een cola*
Interview on NRC.nl (in Dutch, December 9, 2016) available at:
https://www.nrc.nl/advertentie/deloitte/als-het-kouder-wordt-zakt-ook-de-prijs-van-een-cola

– *Dynamic pricing in retail: Developments from science and practice*
Blog post on Deloitte.com (December 8, 2016) available at:
https://www.deloitte.com/nl/nl/pages/data-analytics/articles/dynamic-pricing-in-retail.html

– *Dynamic pricing: A win-win proposition*
Blog post on Deloitte.com (July 14, 2017) available at:
https://www.deloitte.com/nl/nl/pages/data-analytics/articles/dynamic-pricing.html

– *Accountability of Artificial Intelligence: how to see what is inside the black box?*
Blog post on Deloitte.com (May 7, 2018) available at:
https://www.deloitte.com/nl/nl/pages/data-analytics/articles/accountability-of-artificial-intelligence-how-to-see-what-is-inside-the-black-box.html

Chapter 2

Price Optimization under the Finite-Mixture Logit Model

2.1 Introduction

2.1.1 Background and motivation

For any business to be sustainably competitive it is of crucial importance to ask the *right* price for the products that it sells. This is notoriously challenging, especially when taking into account that each customer values the products differently, and that a price change in one particular product also changes the demand for the other products in the assortment—so-called substitution of demand across products. To give substance to this problem, customer choice models have been adopted to describe substitution patterns across products, and optimize prices accordingly. These choice models assume that customers assign utility to products based on the products' attributes, and purchase the product that maximizes their utility (or, possibly, choose not to purchase after all). Besides its theoretical relevance, using customer choice models is practically appealing, given that the increased availability of customer-level purchasing data allows for its adoption in business practice.

Existing work has considered customer choice models—most notably the multinomial logit (MNL) and the nested logit model—that assume customers are homogeneous. This means that the probability with which a specific product is purchased, is constant across customers. Stated differently: any two customers purchase any particular product with the same probability. We consider a pricing problem that departs from this stringent assumption of customer homogeneity. Customer heterogeneity is incorporated by as-

suming that each customer belongs to one of a finite number of customer segments, and that customers choose according to a multinomial logit (MNL) model with segment-specific parameters. The resulting model is known as the finite-mixture logit model (FML). Our algorithm runs in polynomial time in the number of products, and obtains prices at which the revenue is guaranteed to be at least $(1-\epsilon)$ times the maximum attainable revenue for any prespecified $\epsilon > 0$.

The revenue function induced by the FML choice model is not quasiconcave in general and possibly multimodal. Consequently, to obtain a $(1-\epsilon)$-approximation with existing global optimization methods requires exponential time in the number of products (see, e.g., Pintér, 2013). Such solvers typically exploit (local) Lipschitz constants of the revenue function to systematically branch and bound subsets of the solution space. However, this approach leads to prohibitively long computation times when the assortment is larger than just a handful of products—Fowkes et al. (2013) and Cartis et al. (2015), for example, reveal that global optimization algorithms can be used to solve problems up to about four dimensions, after which computation times surge due to the curse of dimensionality.

The running time of our algorithm, however, is exponential in the number of customer segments, but only polynomial in the number of products. This is of great practical value, since the number of products that a retailer sells is potentially very large. Meanwhile, empirical studies on customer choice behavior in retail and transportation find that a low number of segments suffices to capture heterogeneity appropriately—Chintagunta et al. (1991) and Wen and Lai (2010) use two segments, Greene and Hensher (2003), Shen (2009), and Bhat (1997) use three segments, and Scarpa and Thiene (2005) use four segments. In revenue management settings, usually two segments—the business and the leisure segment—are used to incorporate customer heterogeneity. This formed the basis for the seminal paper of Littlewood from 1972, in which he established what became known as Littlewood's rule (see Littlewood, 2005, for a reprint). More recently, Hetrakul and Cirillo (2015) and Delahaye et al. (2017) used two segments in an empirical study on customer choice behavior in the airline industry and railway services, respectively.

2.1.2 Literature review

The MNL model, which is a special case of FML, is arguably the most famous customer choice model and has received tremendous attention amongst both scholars and practitioners since it was introduced by Luce (1959). It was first considered in the con-

text of price optimization by Hanson and Martin (1996), who show by counter-example that the profit function under MNL is not quasiconcave, and provide a path-following heuristic for numerical optimization. In Song and Xue (2007), Dong et al. (2009), and Li and Huh (2011) it is shown that the profit function under MNL as a function of the purchase probabilities (or, equivalently, the market shares) is concave. This allows for efficient maximization with respect to the purchase probabilities, which have a one-to-one correspondence with the prices. It is also shown (Dong et al., 2009, Corollary 2) that the optimal markup, i.e., the optimal price minus the marginal costs, is constant across products (see also Proposition 5.1 in Aydin and Ryan (2000), Lemma 7 in Maddah and Bish (2007), Theorem 6 in Akcay et al. (2010), and Proposition 1 in Wang (2012)). However, the constant markup property only holds when inventory is sufficient to meet all demand at all times (Dong et al., 2009, Akcay et al., 2010) and when the price sensitivities are constant across products (Li and Huh, 2011). In Aydin and Porteus (2008) it is shown that the first-order conditions of the revenue function under MNL are sufficient for global optimality and, in case of constant price sensitivities, closed-form solutions are provided by Akcay et al. (2010).

One of the major drawbacks of MNL is that it assumes a very specific substitution pattern, known as the independence under irrelevant alternatives (IIA) property. The IIA property implies that for any two alternatives, the ratio of the probability of choosing one over the probability of choosing the other is independent of the composition or the attributes of the other alternatives. In the case of price optimization, this implies that an increase in the price of one product increases the purchase probabilities of all other products with the same proportion.

Other choice models have been considered in the literature that alleviate the IIA property (but do assume customer homogeneity). The most prominent is the nested logit (NL) model, which generalizes MNL by assuming that customers first choose a *nest* (a subset) of products, and subsequently choose a product from that particular nest according to an MNL model. For the NL model, it holds that the profit is concave in the purchase probabilities when the price sensitivities are constant within nests (Li and Huh, 2011). In the more general case of product-specific price sensitivities, Gallego and Wang (2014) show amongst other things that the adjusted markup, which is defined as the optimal price minus the marginal costs minus the reciprocal of the price sensitivity, is constant for all the products within a nest. For price and assortment optimization under NL with price bounds, Rayfield et al. (2015) provide numerical approximation methods. Finally, Li et al. (2015) and Huh and Li (2015) consider a more general version of the NL model in which nests can contain nests of products, leading to a tree structure of nests of arbitrary depth. Evidently, choosing an appropriate composition of nests can

be challenging in practice (Koppelman and Bhat, 2006).

Recently, other choice models than MNL and NL have been considered in the literature. Alptekinoğlu and Semple (2016) consider the problem of optimal pricing under the exponomial choice model, in which other distributional assumptions than MNL and NL are imposed, which suit certain applications better. Li and Webster (2017a) consider the paired combinatorial logit model. This alternative to NL models choice settings in which the sequence of choices, as modeled in NL by the nested structure, is not apparent. Zhang et al. (2018) study pricing problems when customers choose according to any model from the family of generalized extreme value choice models, which includes MNL, NL, and the paired combinatorial logit model. In this general setting, Zhang et al. (2018) derive a closed-form solution for optimal prices in case the price sensitivity is constant across products. Finally, Dong et al. (2018) consider the pricing problem when customers choose according to the Markov chain choice model, in which customers transition between products until a product is purchased or the no-purchase option is reached. Amongst other things, Dong et al. (2018) show in a monopolistic setting how to obtain the optimal prices efficiently.

In all of the aforementioned references, except Hanson and Martin (1996), customers are assumed to be homogeneous: the expected utility for a particular product is assumed to be constant across customers. Consequently, any two customers purchase a particular product with the same probability. This is problematic, since the consequences of ignoring heterogeneity when it is present are potentially severe, given that it may lead to biased parameter estimates (see, e.g., Hsiao, 2014, p. 10) and consequently to sub-optimal prices. Whilst the importance of incorporating customer heterogeneity in choice models has long been acknowledged (Guadagni and Little, 1983), progress on price optimization under models that allow for customer heterogeneity has been limited.

The FML model, which is considered in this chapter, overcomes the restrictive assumptions of a homogeneous population and IIA by assuming that customers belong to one of a number of customer segments. Each of these customer segments chooses according to an MNL model with segment-specific parameters. As such, FML belongs to the family of mixed logit models, which are flexible enough to approximate any customer choice model up to any level of desired accuracy (McFadden et al., 2000). The literature on pricing under FML is sparse—the only two references that we are aware of are Hanson and Martin (1996) and Li and Webster (2017b). The former provides no structural results nor (global) optimality guarantees on the solution obtained. The latter shows that, in general, the profit function under FML as a function of the purchase

probabilities is not (quasi)concave. The paper provides an efficient algorithm in case the revenue function is quasiconcave, but no conditions are provided that ensure quasiconcavity of the revenue function, which limits the applicability of their algorithm. For non-quasiconcave revenue functions a gradient-descent approach is proposed, but bounds on the quality of the obtained solution are not provided.

2.1.3 Contributions

We propose an algorithm that optimizes revenue with respect to prices when customers choose according to the FML customer choice model. Our algorithm obtains prices at which the revenue is guaranteed to be at least $(1 - \epsilon)$ times the maximum attainable revenue, for any prespecified $\epsilon > 0$. The running time is exponential in the number of customer segments, but only polynomial in the number of products. This is beneficial, since typically a small number of customer segments suffices to capture customer heterogeneity appropriately (see references in Section 2.1.1), whilst the number of products in the assortment of a retailer is potentially very large. Our numerical experiments show that the algorithm performs very well in practice—the computation times vary from a few seconds to minutes on a single CPU.

To develop our algorithm, we derive a novel characterization of the problem of optimizing prices under FML. The approach is based on a reparameterization of the original revenue maximization problem, in which we treat the no-purchase probability of each segment—the probability that a customer from a particular segment does not purchase anything—as a parameter. We show that, when equality constraints are imposed on the no-purchase probabilities, the problem of optimizing prices can be formulated as a concave maximization problem with linear constraints. Using this result, we reduce the initial problem of finding the optimal prices, to the problem of finding the optimal no-purchase probabilities. This results in a dimension reduction in case the number of segments is smaller than the number of products.

To optimize revenue with respect to prices under equality constraints on the no-purchase probabilities, we formulate the Lagrangian and provide a closed-form solution for the dual problem. Using this dual problem, we derive a local Lipschitz constant of the revenue as a function of the no-purchase probabilities. In addition, we formulate a linear program that verifies whether a given vector is in the set of attainable no-purchase probabilities—that is, the linear program verifies if prices exist that induce a vector of no-purchase probabilities equal to this given vector.

These results are leveraged to obtain a branch-and-bound algorithm that systematically

searches the space of all possible no-purchase probabilities. The algorithm terminates when the obtained solution meets the desired accuracy, and is guaranteed to terminate in a finite number of steps. The running time grows only polynomially in the number of products, and is shown to run fast in numerical experiments. As such, our work improves the applicability of price optimization in business practice, where the importance of accounting for customer heterogeneity has long been acknowledged.

2.1.4 Outline

The rest of this chapter is organized as follows. In Section 2.2, we formulate the revenue optimization problem under the FML choice model. Then, in Section 2.3, we develop a new characterization of this problem and provide our proposed optimization algorithm. Section 2.4 contains the results of our numerical experiments. An alternative approach to optimize prices under FML is presented in Section 2.5. Detailed mathematical proofs are contained in Section 2.A.

2.2 Problem Description

We consider the problem of optimizing the expected revenue with respect to prices for a range of products $N := \{1, 2, \ldots, n\}$, when the population of customers consists of customer segments $M := \{1, 2, \ldots, m\}$, for positive integers n and m. The share of customer segment $c \in M$ equals $w_c \in (0, 1)$ with $\sum_{c \in M} w_c = 1$, i.e., a randomly selected customer belongs to segment c with probability w_c. Customers choose according to an MNL model with segment-specific parameters. More precisely, a customer from segment $c \in M$ assigns (random) utility $U_{ci}(p_i)$ to product $i \in N$ at price $p_i \in \mathbb{R}$, given by

$$U_{ci}(p_i) := a_{ci} - b_i p_i + \varepsilon_{ci}, \tag{2.1}$$

where ε_{ci} is i.i.d. standardized Gumbel distributed (i.e., $\mathbb{P}(\varepsilon_{ci} \leq x) = \exp(-\exp(-x))$ for all $x \in \mathbb{R}$), $a_{ci} \in \mathbb{R}$ is the baseline product valuation by customers from segment c for product i, and $b_i \in \mathbb{R}_{++}$ is the product-specific price sensitivity; here, and in the rest of this work, we denote $\mathbb{R}_+ := [0, \infty)$ and $\mathbb{R}_{++} := (0, \infty)$. We set $U_{c0} := \varepsilon_{c0}$ with ε_{c0} i.i.d. standardized Gumbel distributed as the utility for the outside option, i.e., the utility derived from not buying any product from the product set N. This is a standard approach in customer choice modeling to ensure that the parameters in the model are identified when the model's parameters are estimated from data (Train, 2009).

Then, the probability that a customer from segment $c \in M$ purchases product $i \in N$ under prevailing prices $p = (p_1, \ldots, p_n) \in \mathbb{R}^n$ equals

$$q_{ci}(p) := \mathbb{P}\left\{\arg\max_{j \in N} U_{cj}(p_j) = i\right\} = \frac{\exp\left(a_{ci} - b_i p_i\right)}{1 + \sum_{j \in N} \exp\left(a_{cj} - b_j p_j\right)},$$

which is well defined, since ties in utility occur with probability zero (see Train (2009) for a discussion and a derivation of the choice probabilities). Furthermore,

$$q_{c0}(p) := \frac{1}{1 + \sum_{j \in N} \exp\left(a_{cj} - b_j p_j\right)}$$

denotes the no-purchase probability of segment $c \in M$, i.e., the probability that a customer of segment c does not purchase anything. Hence, the total expected revenue per customer equals

$$\Pi(p) := \sum_{c \in M} w_c \sum_{j \in N} p_j q_{cj}(p),$$

and the optimization problem

$$\max_{p \in \mathbb{R}^n} \Pi(p) \qquad \texttt{(max-rev)}$$

is the problem of interest in this chapter. In the following section we propose our solution method to this problem.

2.3 Solution Method

In general, `max-rev` lacks any useful structure that can be leveraged to solve it to optimality efficiently. For example, the revenue function Π is in general not (quasi)concave in prices and possibly multimodal—for numerical examples of this see Li and Webster (2017b). By relying on standard numerical optimization methods, such as gradient descent, one might converge to a sub-optimal local maximum, without being able to assess the quality of the obtained solution. In this section, we present a solution method for `max-rev` that provides quality guarantees on the obtained solution.

2.3.1 Outline

First, in Section 2.3.2, we show that the problem can be reparameterized so that the revenue can be optimized with respect to the no-purchase probabilities, instead of with

respect to prices. More precisely, we show that, when we impose equality constraints on the no-purchase probabilities, and reparameterize appropriately, the problem of maximizing revenue with respect to prices is a concave maximization problem. Then, in Section 2.3.3, we show that for the aforementioned reparameterized and constrained problem, strong duality holds. Also, we provide a closed-form solution for the dual problem. As a result, for any m-vector of feasible no-purchase probabilities, we are able to maximize the revenue over all prices such that the no-purchase probabilities equal this m-vector. Here, a vector $x \in (0,1)^m$ of no-purchase probabilities is feasible, if a price vector $p \in \mathbb{R}^n$ exists such that $q_{c0}(p) = x_c$ for all $c \in M$. In Section 2.3.4, we derive a local upper bound on the attainable revenue over subsets of no-purchase probabilities. More precisely, for any m-dimensional hypercube in $(0,1)^m$ that contains feasible no-purchase probabilities, we derive an upper bound on the revenue, under the constraint that the no-purchase probabilities are in that hypercube. Finally, in Section 2.3.5, we formulate a linear program to verify for any m-dimensional hypercube in $(0,1)^m$, whether it contains any feasible no-purchase probabilities. If this is the case, the linear program provides a feasible vector of no-purchase probabilities that is in this hypercube.

We use all these ingredients to construct a branch-and-bound algorithm in Section 2.3.6. This algorithm systematically searches for the no-purchase probabilities in $(0,1)^m$ that correspond with the optimal prices. We show in Theorem 2.1 that the algorithm terminates in finite steps an provides a $(1-\epsilon)$-approximation for `max-rev`. In addition, we show in Theorem 2.2 that the time complexity of this algorithm is exponential in m, but only polynomial in n.

2.3.2 Problem reparameterization

In the following proposition we show that a global maximizer of Π exists, and we provide lower and upper bounds on global maximizers of Π.

Proposition 2.1. *A global maximizer of* Π *exists, and for any global maximizer* $p^\star$*, it holds that*

$$\underline{p}_j \leq p_j^\star \leq \bar{p}_j, \text{ for all } j \in N,$$

where

$$\underline{p}_j := b_j^{-1} \text{ and } \bar{p}_j := b_j^{-1} + \max_{c \in M} \max_{p \in \mathbb{R}^n} \sum_{i \in N} p_i q_{ci}(p), \text{ for all } j \in N.$$

Sketch of proof (detailed proof in Appendix 2.A.1). First, to show existence of a global maximizer, we construct a compact set in the domain of Π that must contain all maximizers (if any exist). Then, since Π is bounded and continuous on this compact set, by the

extreme value theorem a global maximizer exists. Then, we show that maximizers only occur in the interior of this compact set, which means that maximizers are attained at stationary points. Finally, based on the first-order conditions, we derive the bounds stated in the proposition. □

Remark 2.1. To compute $\bar{p}_j$ for all $j \in N$, as defined in Proposition 2.1, we require a solution to the optimization problem $\max_{p\in\mathbb{R}^n} p_j q_{cj}(p)$ for all $c \in M$. This problem can be solved efficiently, since, for each $c \in M$, $\max_{p\in\mathbb{R}^n} p_j q_{cj}(p)$ is the unique solution to a univariate fixed-point equation (Theorem 2, Li and Huh, 2011).

Using Proposition 2.1, we reformulate `max-rev` as follows. For $p \in \mathbb{R}^n$, let

$$q_0(p) := [q_{10}(p), \ldots, q_{m0}(p)]^\top,$$

and define the set

$$X := \left\{ q_0(p) : p \in [\underline{p}, \bar{p}] \right\}, \tag{2.2}$$

where $\underline{p}$ and $\bar{p}$ are the column vectors with elements $\underline{p}_j$ and $\bar{p}_j$, respectively, for $j = 1, \ldots, n$. Then, we formulate `max-rev` as the following two-stage problem,

$$\begin{aligned} \max_{x\in X} \quad & \max_{p\in[\underline{p},\bar{p}]} \quad \Pi(p) \\ & \text{such that} \quad q_0(p) = x. \end{aligned} \tag{2.3}$$

Thus, the inner problem in (2.3) maximizes the expected revenue with respect to prices under the constraint that the no-purchase probabilities equal $x \in X$, and the outer problem maximizes the revenue over the set of no-purchase probabilities X. We now show that the inner problem in (2.3) can be reformulated as a concave maximization problem with linear constraints. To this end, let

$$s_{cj} := e^{a_{cj}} \text{ for } c \in M, j \in N,$$

and let

$$k_j := b_j^{-1} \left[w_1 s_{1j}, \ldots, w_m s_{mj} \right]^\top, \text{ for all } j \in N.$$

We change the decision variables from p_j to $z_j = e^{-p_j b_j}$ for all $j \in N$, and we define $z := [z_1, \ldots, z_n]^\top$. Correspondingly, let $\underline{z}_j := e^{-\bar{p}_j b_j}$ and $\bar{z}_j := e^{-\underline{p}_j b_j}$ be lower and upper bounds, respectively, on z_j, for all $j \in N$. In addition, let $\underline{z} := [\underline{z}_1, \ldots, \underline{z}_n]^\top$ and $\bar{z} := [\bar{z}_1, \ldots, \bar{z}_n]^\top$. The following lemma presents a concave reparameterization of the inner maximization problem of (2.3).

Lemma 2.1. *For all $x \in X$, the maximization problem*

$$\max_{p \in [\underline{p}, \bar{p}]} \Pi(p)$$
$$\text{such that } q_{c0}(p) = x_c, \ \forall c \in M$$

is equivalent to

$$\max_{z \in [\underline{z}, \bar{z}]} -\sum_{j \in N} z_j \log(z_j) k_j^\top x$$
$$\text{such that } 1 + \sum_{j \in N} s_{cj} z_j = \frac{1}{x_c}, \ \forall c \in M,$$

which is a concave maximization problem with linear constraints.

Sketch of proof (detailed proof in Appendix 2.A.1). The equivalence of the two maximization problems follows from a change of variables from p_j to $z_j = e^{-p_j b_j}$ for all $j \in N$, and a number of monotonic transformations. The problem is concave since the function $-\log(y)y$ is concave on $y \in (0, e^{-1}]$, $k_j^\top x > 0$ for all $x \in X$ and $j \in N$, and $z_j \in (0, e^{-1}]$, for all $j \in N$ (this follows from the fact that $z_j \in [\underline{z}_j, \bar{z}_j]$, and $0 < \underline{z}_j < \bar{z}_j < e^{-1}$, for all $j \in N$). Clearly, the constraints are linear. □

Lemma 2.1 shows that, after imposing equality constraints on the no-purchase probabilities, the problem of maximizing revenue can be formulated as a concave maximization problem. For further reference, we formally define the concave maximization problem from Lemma 2.1 as follows. For $x \in X$,

$$\pi(x) := \max_{z \in [\underline{z}, \bar{z}]} -\sum_{j \in N} z_j \log(z_j) k_j^\top x \qquad (\texttt{cnstr-rev}(x))$$
$$\text{such that } 1 + \sum_{j \in N} s_{cj} z_j = \frac{1}{x_c}, \quad \forall c \in M.$$

Here, $\texttt{cnstr-rev}(x)$ denotes the maximization problem (`cnstr-rev` is an abbreviation for "constrained revenue") and $\pi(x)$ denotes the maximum attainable revenue when the no-purchase probabilities are constrained to $x \in X$. It is a direct consequence of Lemma 2.1 that

$$\max_{x \in X} \pi(x) = \max_{p \in [\underline{p}, \bar{p}]} \Pi(p). \tag{2.4}$$

In fact, a $(1-\epsilon)$-approximation of the revenue function in the no-purchase probabilities has a one-to-one correspondence with a $(1-\epsilon)$-approximation of the revenue function in prices, for each $\epsilon > 0$, which we show in the following proposition.

Proposition 2.2. *Let $x' \in X$, let $z' \in [\underline{z}, \bar{z}]$ be a maximizer of* `cnstr-rev`(x')*, and let $\epsilon > 0$. If*

$$\pi(x') \geq (1-\epsilon) \max_{x \in X} \pi(x),$$

then, for $p' := \left[-b_1^{-1} \log z_1', \ldots, -b_n^{-1} \log z_n'\right]^\top$, it holds that $p' \in [\underline{p}, \bar{p}]$, and

$$\Pi(p') \geq (1-\epsilon) \max_{p \in [\underline{p}, \bar{p}]} \Pi(p).$$

Sketch of proof (detailed proof in Appendix 2.A.1). Let $x' \in X$ and $\epsilon > 0$ be as in the statement of the proposition and let $z' \in [\underline{z}, \bar{z}]$ be a solution to `cnstr-rev`(x'). First, we show that $p' \in [\underline{p}, \bar{p}]$ by construction of `cnstr-rev`(x'). Then, we show by a number of monotonic transformations that $\Pi(p') = \pi(x')$. Finally, we use Equation (2.4) to obtain the claimed inequality. □

This proposition implies that, to obtain a $(1-\epsilon)$-approximation for the maximum attainable revenue under the FML choice model, it suffices to obtain a $(1-\epsilon)$-approximation to $\max_{x \in X} \pi(x)$. In the following sections, we develop an algorithm to obtain such an approximation efficiently.

2.3.3 Strong duality

We proceed by taking the Lagrangian of `cnstr-rev`(x) for $x \in X$ and, subsequently, constructing the dual. This allows for efficient evaluation of $\pi(x)$, for all $x \in X$, as well as for computing upper bounds on π over subsets of X, which we leverage in our proposed algorithm.

For $x \in X$, let

$$\mu(x) := \left[\frac{1-x_1}{x_1}, \ldots, \frac{1-x_m}{x_m}\right]^\top,$$

and let $S \in \mathbb{R}^{m \times n}$ be the matrix with elements s_{cj} for all $c \in M$ and $j \in N$. This allows us to write the m equality constraints in `cnstr-rev`(x) as $Sz = \mu(x)$, and define the Lagrangian of `cnstr-rev`(x) by

$$\mathcal{L}(z, x, \lambda, \underline{\nu}, \bar{\nu}) := -\sum_{j \in N} z_j \log(z_j) k_j^\top x + \lambda^\top (Sz - \mu(x)) - \underline{\nu}^\top (z - \underline{z}) + \bar{\nu}^\top (z - \bar{z}),$$

where $z \in (0,\infty)^n$ and $x \in X$, and where $(\lambda, \underline{\nu}, \bar{\nu}) \in \mathbb{R}^m \times \mathbb{R}^n \times \mathbb{R}^n$ are the Lagrange multipliers. Furthermore, define

$$\tilde{z}_j(x, \lambda, \underline{\nu}, \bar{\nu}) := \exp\left(\frac{\lambda^\top s_j + \bar{\nu}_j - \underline{\nu}_j}{k_j^\top x} - 1\right), \tag{2.5}$$

where $x \in X$, $(\lambda, \underline{\nu}, \bar{\nu}) \in \mathbb{R}^m \times \mathbb{R}^n \times \mathbb{R}^n$, and $s_j := [s_{1j}, \ldots, s_{mj}]^\top \in \mathbb{R}^m$ is the j^{th} column of S, for all $j \in N$. The following proposition concerns duality of `cnstr-rev`(x).

Proposition 2.3. *For all $x \in X$ and $(\lambda, \underline{\nu}, \bar{\nu}) \in \mathbb{R}^m \times \mathbb{R}^n \times \mathbb{R}^n$, the dual of* `cnstr-rev`$(x)$ *equals*

$$\begin{aligned} \mathcal{D}(x, \lambda, \underline{\nu}, \bar{\nu}) &:= \max_{z \in \mathbb{R}^n_+} \mathcal{L}(z, x, \lambda, \underline{\nu}, \bar{\nu}) && (2.6) \\ &= \sum_{j \in N} \tilde{z}_j(x, \lambda, \underline{\nu}, \bar{\nu}) k_j^\top x - \lambda^\top \mu(x) + \underline{\nu}^\top \underline{z} - \bar{\nu}^\top \bar{z}, && (2.7) \end{aligned}$$

and strong duality holds:

$$\min_{(\lambda, \underline{\nu}, \bar{\nu}) \in \mathbb{R}^m \times \mathbb{R}^n_+ \times \mathbb{R}^n_+} \mathcal{D}(x, \lambda, \underline{\nu}, \bar{\nu}) = \pi(x). \tag{2.8}$$

In addition, the dual problem on the left-hand side of (2.8) *has a unique minimizer.*

Sketch of proof (detailed proof in Appendix 2.A.2). The proof is obtained by observing that $\mathcal{L}$ is concave in z, so that we can maximize $\mathcal{L}$ with respect to z by solving the first-order conditions with respect to z. Strong duality holds by Slater's condition (Boyd and Vandenberghe, 2004, p. 226), since the objective function in `cnstr-rev`(x) is concave, and the constraints are affine. We show unicity of a solution to the left-hand side of (2.8) by showing that $\mathcal{D}$ is strictly convex in $(\lambda, \underline{\nu}, \bar{\nu})$ for all $x \in X$. □

2.3.4 Local revenue bounds

In this section, we derive an upper bound on π over subsets of X. This will serve as an input in the branch-and-bound algorithm proposed in Section 2.3.6.

Denote by $\|\cdot\|$ the infinity norm, i.e., $\|v\| = \max_{i \in \{1,\ldots,n\}} |v_i|$ for $v \in \mathbb{R}^n$ and, correspondingly, denote by $B(x, r)$ the closed infinity-norm ball (or hypercube) with center x and radius $r > 0$: $B(x, r) = \{y : \|x - y\| \leq r\}$. For $x \in X$ and $r > 0$ such that $B(x, r) \subset (0, 1)^m$, we derive an upper bound on $\max_{x' \in B(x,r) \cap X} \pi(x')$ by constructing a local Lipschitz constant of the dual $\mathcal{D}$ as a function of x. We formalize this as follows.

For $x \in X$, let

$$(\lambda^x, \underline{\nu}^x, \bar{\nu}^x) := \underset{(\lambda,\underline{\nu},\bar{\nu})\in\mathbb{R}^m\times\mathbb{R}^n_+\times\mathbb{R}^n_+}{\arg\min} \mathcal{D}(x, \lambda, \underline{\nu}, \bar{\nu}) \tag{2.9}$$

the solution to the dual problem (2.8) (note that (2.9) is well defined by Proposition 2.3), and for $x \in X$ and $r > 0$ let

$$L(x,r) := \max_{c\in M} \max \left\{ \frac{\lambda_c^x}{(x_c - \operatorname{sgn}(\lambda_c^x)\cdot r)^2} + w_c \sum_{j\in N} \frac{s_{cj}}{b_j} \phi(\tilde{z}_j(x + r, \lambda^x, \underline{\nu}^x, \bar{\nu}^x)), \right.$$
$$\left. \frac{-\lambda_c^x}{(x_c + \operatorname{sgn}(\lambda_c^x)\cdot r)^2} - w_c \sum_{j\in N} \frac{s_{cj}}{b_j} \phi(\tilde{z}_j(x - r, \lambda^x, \underline{\nu}^x, \bar{\nu}^x)) \right\}, \tag{2.10}$$

where $\phi(y) := -y\log(y)$, $y \in \mathbb{R}_{++}$, and $\operatorname{sgn}(\cdot)$ denotes the sign operator. Furthermore, for $x \in (0,1)^m$ and $(\lambda, \underline{\nu}, \bar{\nu}) \in \mathbb{R}^m \times \mathbb{R}^n \times \mathbb{R}^n$, denote by $\nabla\mathcal{D}(x, \lambda, \underline{\nu}, \bar{\nu})$ the gradient of $\mathcal{D}$ with respect to the variable x, i.e., with respect to the first argument, evaluated at the point $(x, \lambda, \underline{\nu}, \bar{\nu})$. The following lemma establishes that $L(x,r)$ is a local Lipschitz constant of $\mathcal{D}$.

Lemma 2.2. *Let $x \in X$ and $r > 0$ such that $B(x,r) \subset (0,1)^m$. Then,*

$$\max_{\tilde{x}\in B(x,r)} \|\nabla\mathcal{D}(\tilde{x}, \lambda^x, \underline{\nu}^x, \bar{\nu}^x)\| \le L(x,r). \tag{2.11}$$

Sketch of proof (detailed proof in Appendix 2.A.3). To establish (2.11), we first derive an expression for $\nabla\mathcal{D}$ by differentiating $\mathcal{D}$ with respect to its first argument. Then, we observe that each element of $\nabla\mathcal{D}$ is the sum of monotone functions in x, which allows us to bound the gradient accordingly. □

Let $\bar{s} := \max_{c\in M, j\in N} s_{cj}$, $\underline{b} := \min_{j\in N} b_j$, let $\underline{\sigma}$ be the smallest singular value of S, and let $W : \mathbb{R}_+ \to \mathbb{R}_+$ be the Lambert function (i.e., the inverse of ye^y). For $x \in X$ and $r > 0$ such that $B(x,r) \subset (0,1)^m$, let

$$K(x,r) := n\bar{s}\underline{b}^{-1} \left(\max_{c\in M} \frac{x_c + r}{x_c - r} + \frac{n^{-1/2}\underline{b}^{-1} W(n\bar{s})}{\underline{\sigma} \min_{c\in M} x_c(x_c - r)} \right)$$

and

$$\pi^u(x,r) := \pi(x) + r \cdot (L(x,r) \wedge K(x,r)). \tag{2.12}$$

The following proposition establishes that (2.12) is an upper bound on π over $B(x,r) \cap X$, for all $x \in X$ and $r > 0$ such that $B(x,r) \subset (0,1)^m$.

Proposition 2.4. *Let $x \in X$ and $r > 0$ such that $B(x,r) \subset (0,1)^m$. Then,*

$$\max_{\tilde{x} \in B(x,r) \cap X} \pi(\tilde{x}) \leq \pi^u(x,r). \tag{2.13}$$

Sketch of proof (detailed proof in Appendix 2.A.3). Let $x \in X$ and $r > 0$ such that $B(x,r) \subset (0,1)^m$. First, suppose $L(x,r) \leq K(x,r)$. By strong duality (Proposition 2.3), it follows that $\pi(\tilde{x}) = \mathcal{D}(\tilde{x}, \lambda^{\tilde{x}}, \underline{\nu}^{\tilde{x}}, \bar{\nu}^{\tilde{x}})$ for all $\tilde{x} \in B(x,r) \cap X$. Then, by using weak duality—essentially relaxing the Lagrange multipliers—we bound π from above over $B(x,r) \cap X$. Finally, we observe that $\mathcal{D}$ is Lipschitz continuous in its first argument, with Lipschitz constant $L(x,r)$. This allows us to bound the dual from above over $B(x,r) \cap X$ using Lemma 2.2. Now, suppose $K(x,r) \leq L(x,r)$. For each $\tilde{x} \in B(x,r) \cap X$, we construct an arc that is completely contained in $B(x,r) \cap X$ and connects x with $\tilde{x}$. Then, we show that π is differentiable over this arc and show that $K(x,r)$ is an upper bound on the absolute value of the derivative of π. Finally, we use the mean-value theorem to obtain the claimed upper bound. □

Remark 2.2. In practice, for $x \in X$ and $r > 0$ such that $B(x,r) \subset (0,1)^m$, in (2.12) it virtually always holds that $L(x,r) < K(x,r)$. The purpose of the function K is to show (in Section 2.3.6) that the running time of our proposed algorithm grows polynomially in n.

2.3.5 Feasibility

In this section, we show how to verify, for a given $x \in (0,1)^m$, whether $x \in X$, i.e., whether the inner problem in (2.3) is feasible. Recall that $x \in X$ if a solution $p \in [\underline{p}, \bar{p}]$ exists to the system of non-linear equations $q_0(p) = x$. This condition is by no means trivial to verify. We illustrate X by means of the following example, which also reveals that X is non-convex in general.

Example 1 Consider the case with two segments and four products, i.e., $m = 2$ and $n = 4$. Let the utilities of the segments, as defined in (2.1), be given by $(a_{11}, a_{12}, a_{13}, a_{14}) = (1,2,3,4)$, $(a_{21}, a_{22}, a_{23}, a_{24}) = (2,1,2,1)$, and $b = (0.03, 0.02, 0.025, 0.01)$. Let the customer segment shares be $w_1 = 0.5$ and $w_2 = 0.5$. By Proposition 2.1, $\underline{p} = \left[33\frac{1}{3}, 50, 40, 100\right]$ and $\bar{p} \approx [255.01, 271.67, 261.67, 321.67]$. Figure 2.1 illustrates X in the two-dimensional plane, which also shows that X is non-convex.

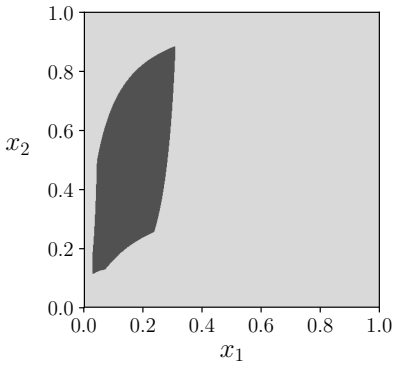


Figure 2.1: The dark area represents X for Example 1.

Our approach is to construct a linear program that verifies, for any given $x \in (0,1)^m$ and $r > 0$ such that $B(x,r) \subset (0,1)^m$, both whether $x \in X$ and whether $B(x,r) \cap X \neq \emptyset$. In case $B(x,r) \cap X \neq \emptyset$, but $x \notin X$, the linear program delivers a vector in $B(x,r) \cap X$. Paraphrased verbally: using our proposed linear program, we can determine for any hypercube in $(0,1)^m$ if the hypercube's center point is feasible. If this is not the case, we can determine if there is another point in the hypercube that is feasible.

Let $x \in (0,1)^m$ and $r > 0$ such that $B(x,r) \subset (0,1)^m$, and consider the following linear program:

$$
\begin{aligned}
\min \quad & \gamma && && (\texttt{LP}(x,r)) \\
\text{such that} \quad & \sum_{j \in N} s_{cj} z_j + \delta_c^+ - \delta_c^- = \mu_c(x), && c = 1,\dots,m && (2.14) \\
& \gamma \geq \delta_c^+ + \delta_c^-, && c = 1,\dots,m && (2.15) \\
& \delta_c^+ \leq \mu_c(x) - \mu_c(x+r), && c = 1,\dots,m && (2.16) \\
& \delta_c^- \leq \mu_c(x-r) - \mu_c(x), && c = 1,\dots,m && (2.17) \\
& \underline{z}_j \leq z_j \leq \bar{z}_j, && j = 1,\dots,n && (2.18) \\
& \gamma \in \mathbb{R} \\
& \delta^+, \delta^- \in \mathbb{R}^m_+,
\end{aligned}
$$

where γ, z, δ^+, and δ^- are the decision variables. The following proposition formalizes the use of $\texttt{LP}(x,r)$.

Proposition 2.5. *Let $x \in (0,1)^m$ and $r > 0$ such that $B(x,r) \subset (0,1)^m$. Then,*

$$B(x,r) \cap X \neq \emptyset \text{ if and only if } \texttt{LP}(x,r) \text{ is feasible.}$$

Sketch of proof (detailed proof in Appendix 2.A.4). Let $x \in (0,1)^m$ and $r > 0$ such that $B(x,r) \subset (0,1)^m$. ($\Rightarrow$) Suppose that $B(x,r) \cap X \neq \emptyset$. To show that $\texttt{LP}(x,r)$ is feasible, we provide a set of decision variables that satisfies all constraints of $\texttt{LP}(x,r)$. ($\Leftarrow$) Suppose that $\texttt{LP}(x,r)$ is feasible. To show that $B(x,r) \cap X \neq \emptyset$, we construct an x^δ such that $x^\delta \in B(x,r) \cap X$. $\square$

In our optimization algorithm, provided in Section 2.3.6, we rely on Proposition 2.5 to determine if a given hypercube in $(0,1)^m$ contains a feasible point, i.e., if it contains an attainable vector of no-purchase probabilities. If this is not the case, we can safely disregard the hypercube, since then it does not contain a maximizer of π. If a given hypercube does contain a feasible vector of no-purchase probabilities, the following corollary establishes how to obtain such a feasible vector.

For $x \in (0,1)^m$ and $\delta^+, \delta^- \in \mathbb{R}^m_+$, define

$$\chi(x, \delta^+, \delta^-) := \left[\frac{x_1}{1 + x_1 \cdot \left(\delta_1^- - \delta_1^+\right)}, \ldots, \frac{x_m}{1 + x_m \cdot \left(\delta_m^- - \delta_m^+\right)} \right]^\top.$$

Corollary 2.1. *Let $x \in (0,1)^m$ and $r > 0$ such that $B(x,r) \subset (0,1)^m$, and suppose that $B(x,r) \cap X \neq \emptyset$. Let $(\gamma, z, \delta^+, \delta^-)$ be a solution to $\texttt{LP}(x,r)$. Then,*

$$\chi(x, \delta^+, \delta^-) \in B(x,r) \cap X. \tag{2.19}$$

If it also holds that $x \in X$, then

$$\chi(x, \delta^+, \delta^-) = x. \tag{2.20}$$

Sketch of proof (detailed proof in Appendix 2.A.4). The proof is a direct consequence of the proof of Proposition 2.5. $\square$

Corollary 2.1 reveals that, if $x \in (0,1)^m$ and $r > 0$ are such that $B(x,r) \cap X \neq \emptyset$, but $x \notin X$, the mapping χ can be used to obtain a feasible point in $B(x,r)$. In the following proposition, we provide an interpretable characterization of χ.

Proposition 2.6. *Let $x \in (0,1)^m$ and $r > 0$ such that $B(x,r) \subset (0,1)^m$ and $B(x,r) \cap X \neq \emptyset$. Let $(\gamma, z, \delta^+, \delta^-)$ be a solution to $\texttt{LP}(x,r)$. Then,*

$$\min_{x' \in B(x,r) \cap X} \|\mu(x) - \mu(x')\| = \|\mu(x) - \mu(\chi(x, \delta^+, \delta^-))\|.$$

Sketch of proof (detailed proof in Appendix 2.A.4). We prove the claim by showing that if a $x' \in B(x,r) \cap X$ exists such that $x' \neq \chi(x, \delta^+, \delta^-)$ and $\|\mu(x) - \mu(x')\| < \|\mu(x) -$

$\mu(\chi(x,\delta^+,\delta^-))\|$, then γ is no minimizer of $\texttt{LP}(x,r)$, leading to contradiction. □

Summarizing, for $x \in (0,1)^m$ and $r > 0$ such that $B(x,r) \subset (0,1)^m$, only if $B(x,r) \cap X \neq \emptyset$ then, by Proposition 2.5, a solution to $\texttt{LP}(x,r)$ exists. Furthermore, for any solution $(\gamma, z, \delta^+, \delta^-)$ to $\texttt{LP}(x,r)$, by Corollary 2.1, $\chi(x,\delta^+,\delta^-) \in B(x,r) \cap X$. In our optimization algorithm, provided in Section 2.3.6, we use these results to determine, for a given hypercube in $(0,1)^m$, if it contains feasible points and, if it does, evaluate the revenue function π at a feasible point in the hypercube to obtain bounds on π in that hypercube.

2.3.6 Branch-and-bound optimization algorithm

In this section, we present a branch-and-bound algorithm that delivers a $(1-\epsilon)$-approximation for the revenue maximization problem when customers choose according to the FML choice model. We refer to this algorithm as $\texttt{BNB}(\epsilon)$. For any $\epsilon > 0$, $\texttt{BNB}(\epsilon)$ delivers a vector of no-purchase probabilities $x' \in X$ such that $\pi(x') \geq (1-\epsilon)\max_{x\in X}\pi(x)$. By Proposition 2.2, this x' corresponds with a $p' \in \mathbb{R}^n$ such that $(1-\epsilon)\Pi(p') \geq \max_{p\in\mathbb{R}^n}\Pi(p)$.

Algorithm description and explanation

We present $\texttt{BNB}(\epsilon)$ in the pseudo-code block Algorithm 1, in which we write

$$\underline{x} := q_0(\underline{p}) \text{ and } \bar{x} := q_0(\bar{p}),$$

which are lower and upper bounds, respectively, on the no-purchase probabilities. We first explain the workings of $\texttt{BNB}(\epsilon)$ by means of Example 1 from Section 2.3.5. Thereafter, we formally establish that the algorithm terminates in finite steps and delivers a $(1-\epsilon)$-approximation for $\epsilon > 0$. Note that, in this example, there are four products, but we only optimize over the two no-purchase probabilities of the two customer segments.

Figure 2.2 illustrates the first six iterations of $\texttt{BNB}(\epsilon)$ for Example 1 with $\epsilon = 0.01$, where one iteration is understood to be one run over the code block under the while clause (i.e., one run over the lines 5-23). We now explain, by means of Figure 2.2, how we initialize $\texttt{BNB}(\epsilon)$, and what one iteration of $\texttt{BNB}(\epsilon)$ comprehends.

Initialization. We initialize $\texttt{BNB}(\epsilon)$ by constructing a hypercube that contains the set of all attainable no-purchase probabilities X. More precisely, $\texttt{BNB}(\epsilon)$ is initialized by

Algorithm 1 $\texttt{BNB}(\epsilon)$

$\underline{\pi} \leftarrow \Pi(\underline{p})$
$\bar{\pi} \leftarrow \infty$
$r \leftarrow \frac{1}{2} \max_{c \in M}(\bar{x}_c - \underline{x}_c)$
$\mathcal{X}_1 \leftarrow \{\underline{x} + r\}$
while $(1-\epsilon)\bar{\pi} > \underline{\pi}$ **do**
    $\bar{\pi} \leftarrow -\infty$
    $\mathcal{X}_2 \leftarrow \emptyset$
    for x in $\mathcal{X}_1$ such that $B(x,r) \cap X \neq \emptyset$ **do** ▷ Check feasibility
        $(\gamma, z, \delta^+, \delta^-) \leftarrow$ a solution to $\texttt{LP}(x,r)$
        $x^\delta \leftarrow \chi(x, \delta^+, \delta^-)$
        $r^\delta \leftarrow r + \|x - x^\delta\|$
        if $\pi(x^\delta) \geq \underline{\pi}$ **then** $\underline{\pi} \leftarrow \pi(x^\delta),\ x^\star \leftarrow x^\delta$ ▷ Update current best solution
        if $B(x^\delta, r^\delta) \subset (0,1)^m$ **then**
            if $\pi^u(x^\delta, r^\delta) \geq \underline{\pi}$ **then** $\mathcal{X}_2 \leftarrow \mathcal{X}_2 \cup \{x\}$ ▷ Stage for branching
            if $\pi^u(x^\delta, r^\delta) \geq \bar{\pi}$ **then** $\bar{\pi} \leftarrow \pi^u(x^\delta, r^\delta)$ ▷ Update upper bound
        else
            $\mathcal{X}_2 \leftarrow \mathcal{X}_2 \cup \{x\}$ ▷ Stage for branching
            $\bar{\pi} \leftarrow \infty$ ▷ Update upper bound
    $r \leftarrow \frac{1}{2} r$
    $\mathcal{X}_1 \leftarrow \emptyset$
    for x in $\mathcal{X}_2$ **do** ▷ Branching
        for ω in $\{-1,1\}^m$ **do**
            $\mathcal{X}_1 \leftarrow \mathcal{X}_1 \cup \{x + \omega r\}$

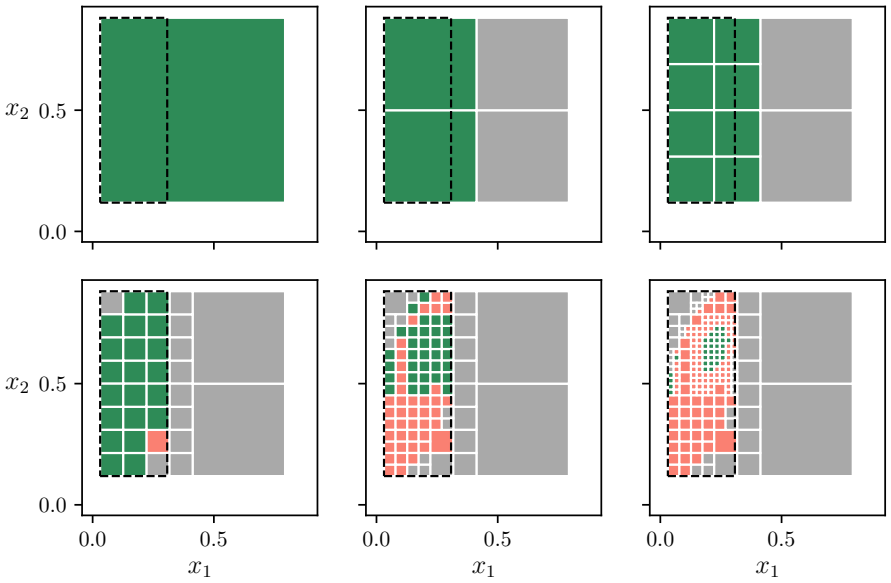


Figure 2.2: The first six iterations of $\texttt{BNB}(\epsilon)$ for Example 1, starting top left. The dashed rectangle in each pane represents $[\underline{x}, \bar{x}]$, the green cubes possibly contain a maximizer, the grey cubes are infeasible, and the red cubes are sub-optimal.

constructing $r \in \mathbb{R}_{++}$ on line 3 such that $B(\underline{x} + r, r)$ is the smallest hypercube that contains $[\underline{x}, \bar{x}]$. Observe that this implies that $B(\underline{x} + r, r) \supseteq X$, since, by definition of X, $\underline{x}$, and $\bar{x}$, and by construction of r, it holds that

$$X \subseteq [\underline{x}, \bar{x}] \subseteq B(\underline{x} + r, r) \subset (0,1)^m. \tag{2.21}$$

In Figure 2.2, $B(\underline{x} + r, r)$ is the green cube in the first pane, and the dashed rectangle represents $[\underline{x}, \bar{x}]$. The variable $\underline{\pi}$ is used to store the current best lower bound on $\max_{x \in X} \pi(x)$ and is initialized with $\Pi(\underline{p})$. In each iteration, $\texttt{BNB}(\epsilon)$ constructs an upper bound on $\max_{x \in X} \pi(x)$, which is stored in $\bar{\pi}$. This variable is initialized with ∞.

Checking feasibility. At the beginning of each iteration at line 5, $r \in \mathbb{R}_{++}$ and $\mathcal{X}_1$ are such that the set $\bigcup_{x \in \mathcal{X}_1} B(x, r)$ contains all maximizers of π (we show this in the proof of Theorem 2.1). On line 8, for each $x \in \mathcal{X}_1$, we verify if $B(x, r) \cap X \neq \emptyset$ by verifying if $\texttt{LP}(x, r)$ is feasible (see Proposition 2.5). Clearly, for each $x \in \mathcal{X}_1$ such that $B(x, r) \cap X = \emptyset$, it holds that $B(x, r)$ does not contain any maximizers—these are the grey cubes in Figure 2.2 and are discarded from the search space.

Bounding revenue. For each $x \in \mathcal{X}$ such that $B(x, r) \cap X \neq \emptyset$, we solve $\texttt{LP}(x, r)$

and construct x^δ and r^δ (on line 10 and 11) such that $x^\delta \in X$ (by Corollary 2.1) and $B(x^\delta, r^\delta) \supseteq B(x, r)$ (by construction of r^δ). Recall that, by Corollary 2.1, $x^\delta = x$ and $r^\delta = r$, if $x \in X$. Then, on line 12, we evaluate π at x^δ. In case $\pi(x^\delta)$ exceeds the lower bound $\underline{\pi}$, we set $\pi(x^\delta)$ as the new lower bound. Subsequently, if $B(x^\delta, r^\delta) \subset (0,1)^m$, we check on line 14 if our upper bound on π over $B(x^\delta, r^\delta) \cap X$, hence over $B(x, r) \cap X$, exceeds the current best lower bound $\underline{\pi}$ (see Proposition 2.4). If this is the case, then $B(x, r)$ possibly contains a maximizer of π, and we store x in $\mathcal{X}_2$ for further consideration. In Figure 2.2, these cubes are colored green. If this is not the case, i.e., if the upper bound on the revenue over $B(x^\delta, r^\delta) \cap X$ does not exceed the current best lower bound, we disregard this cube, i.e., we do not add it to $\mathcal{X}_2$. In Figure 2.2, these cubes are colored green. If for some $x \in \mathcal{X}_1$, it holds on line 13 that $B(x^\delta, r^\delta) \not\subset (0,1)^m$, then we add x to $\mathcal{X}_2$ for further consideration and set the upper bound on the revenue to ∞.

Branching hypercubes. On lines 19-23, for each $x \in \mathcal{X}_2$, i.e., for each hypercube that possibly contains a maximizer of π, $B(x, r)$ is *branched* into 2^m smaller hypercubes of equal size. More precisely, we divide the radius r by two, and for each $x \in \mathcal{X}_2$, we generate 2^m new equidistant centers, and store these in $\mathcal{X}_1$. In Figure 2.2, this can be seen from the fact that the green cubes are partitioned into four smaller cubes in each iteration (and the red and grey cubes not).

Convergence guarantee and computation time

The following theorem establishes that, for $\epsilon > 0$, $\texttt{BNB}(\epsilon)$ obtains a $(1-\epsilon)$-approximation for the revenue maximization problem under the FML choice model.

Theorem 2.1. *For any $\epsilon > 0$, $\texttt{BNB}(\epsilon)$ terminates in a finite number of steps and delivers a $p' \in \mathbb{R}^n$ such that*

$$\Pi(p') \geq (1-\epsilon) \max_{p \in \mathbb{R}^n} \Pi(p). \tag{2.22}$$

Sketch of proof (detailed proof in Appendix 2.A.5). We first derive a finite upper bound on the number of iterations required for $\texttt{BNB}(\epsilon)$ to terminate. Subsequently, we show that, if $\texttt{BNB}(\epsilon)$ terminates, then (2.22) holds by showing that $\texttt{BNB}(\epsilon)$ never discards hypercubes that contain a maximizer of π. □

The following result shows that the running time of BNB grows exponentially in m, but only polynomially in n.

Theorem 2.2. *The time complexity of $\texttt{BNB}(\epsilon)$ is $\mathcal{O}\left(\epsilon^{-m} n^{5.5+3m}\right)$.*

Sketch of proof (detailed proof in Appendix 2.A.5). We show that the total number of hypercubes evaluated by BNB(ϵ) is $\mathcal{O}(\epsilon^{-m} n^{3m})$ and that the time complexity of evaluating a single hypercube is $\mathcal{O}(m^{5.5} n^{5.5})$. The latter follows from the time complexity of solving a linear program. □

Theorem 2.2 is a theoretical result that reveals that the running time of our algorithm grows polynomially in the number of products. In the following section, we provide the results from our numerical study.

2.4 Numerical Study

2.4.1 Preliminaries

In this section, we present a numerical study on the performance of BNB. In particular, we illustrate how the running time of BNB depends on the number of segments (m) and the number of products (n). We implemented BNB in Python 3, and our implementation is publicly available at https://github.com/rubenvdg/fml-pricing.

We solve the linear program LP by using the Python library `cvxopt` (Andersen et al., 2018). To evaluate π on line 12 of BNB, we numerically solve the dual problem (2.8) by using the Broyden-Fletcher-Goldfarb-Shanno algorithm (Nocedal and Wright, 1999), as implemented in the SciPy library (Jones et al., 2001). All the experiments were carried out on macOS using a single 2.5 GHz Intel Core i7 processor. We foresee that substantial improvements with regards to running time can be achieved by using high-performance computers and/or implementing the algorithm in a faster programming architecture, such as C/C++. We like to emphasize that BNB allows for straightforward parallelization by processing the for-loop on line 8 in parallel. Hence, the performance indicators that we present can safely be regarded as conservative.

Each time we run BNB, we randomly sample the parameters of the FML choice model as follows. We sample the segments shares w_c, for all $c \in M$, from the unit interval, and normalize to ensure that $\sum_{c \in M} w_c = 1$. The parameters that concern the utility assigned to the products, as defined in (2.1), are sampled as follows. For all $c \in M$ and $j \in N$, we randomly sample $a_{cj} \sim U(-4, 4)$, where $U(\ell, u)$ denotes the continuous uniform distribution over the interval $[\ell, u]$, for $\ell, u \in \mathbb{R}$. These values are in line with the empirical work of Delahaye et al. (2017) and Li and Webster (2017b), and the simulation framework of Rayfield et al. (2015). We randomly sample the price sensitivities

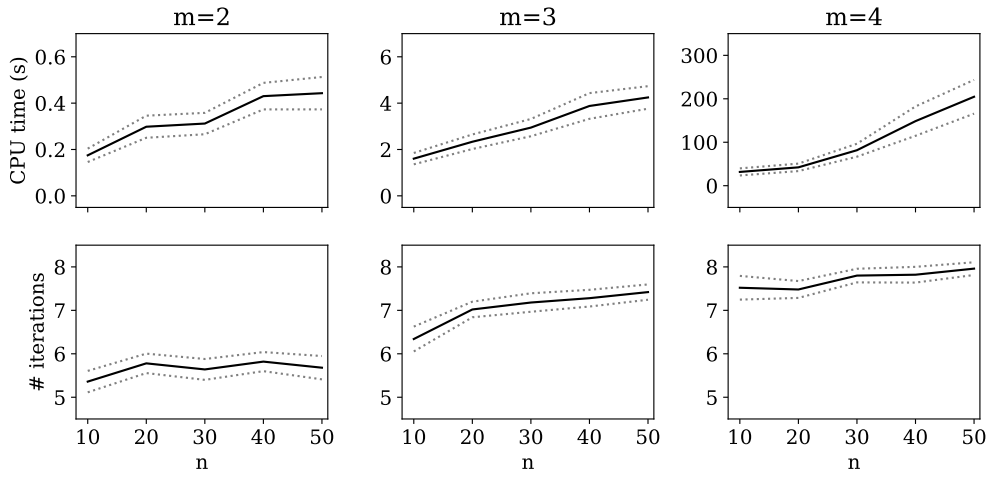


Figure 2.3: The running time in seconds and the number of required iterations with $\epsilon = 0.01$. The dotted lines indicate the 95% confidence interval, based on 50 simulations for each $n \in \{10, 20, \ldots, 50\}$ and $m \in \{2, 3, 4\}$.

$b_j \sim U(-0.001, -0.01)$, for all $j \in N$. This corresponds with a broad range of price sensitivities since, in the most extreme case, customers are ten times more price sensitive for some products than for others.

2.4.2 Results

We set $\epsilon = 0.01$, so that the obtained solution corresponds with expected revenue that is, in the worst case, 99% of the maximum attainable expected revenue (Theorem 2.1). To illustrate how the running time of $\texttt{BNB}(\epsilon)$ grows in the number of products n, we ran $\texttt{BNB}(\epsilon)$ fifty times for each $n \in \{10, 20, \ldots, 50\}$ and $m \in \{2, 3, 4\}$, until the algorithm terminated. In Figure 2.3, we illustrate the results of these simulations. This figure shows the mean number of iterations and the mean CPU computation time (in seconds) required for $\texttt{BNB}(\epsilon)$ to converge, together with a 95% confidence interval for the mean. We observe that the running time grows approximately linear in the number of products. From a practical point of view, $\texttt{BNB}(\epsilon)$ runs very fast on large-scale problems—the average running time varies from approximately 0.2 seconds (for two segments and ten products) to three minutes (for four segments and fifty products). It is worth emphasizing that we only use open-source software and consumer-purpose hardware to obtain these results.

Figure 2.4 illustrates the growth in computation time as the number of segments m

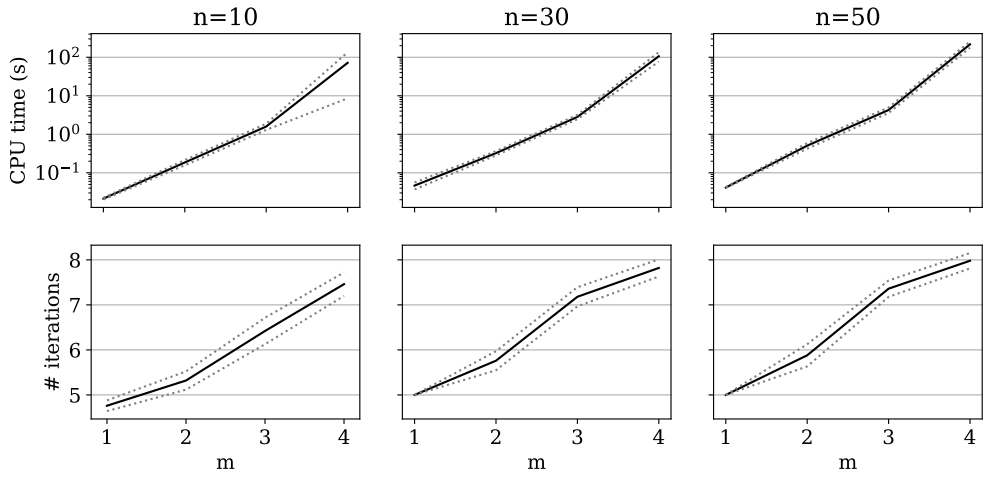


Figure 2.4: The running time in seconds and the number of required iterations, for $m = 1, 2, 3, 4$ and $n = 10, 30, 50$, with $\epsilon = 0.01$. The dotted lines indicate the 95% confidence interval, based on 50 simulations for each $n \in \{5, \ldots, 50\}$ and $m \in \{1, 2, 3, 4\}$.

increases from one to four, for $n \in \{10, 30, 50\}$. To construct Figure 2.4, we ran fifty simulations for each $m \in \{1, 2, 3, 4\}$ and $n \in \{10, 30, 50\}$ until $\texttt{BNB}(\epsilon)$ terminated, with $\epsilon = 0.01$. Note the logarithmic scale of the y-axis of the top three panes in Figure 2.4. Hence, the straight lines in the top three panes indicate that the running time grows exponentially in the number of segments, which is in line with Theorem 2.2.

2.5 Alternative Approach

The approach of this chapter has been to reparameterize Π (the revenue function that takes prices as input) to obtain π (a revenue function that takes the no-purchase probabilities as input). In this section, we present yet another reparameterization (that also takes the no-purchase probabilities as input).

Instead of parameterizing the no-purchase probabilities in the revenue function Π (see Section 2.3), we parameterize the no-purchase probabilities in the first-order conditions of Π. More precisely, we take the derivative of Π with respect to the prices and equate them to zero to obtain the first-order conditions. Then, we parameterize the no-purchase probabilities analogously to Section 2.3.2 and show that, when keeping the no-purchase probabilities constant, a unique vector of prices exists that solves the resulting system of non-linear equations. This leads to yet another revenue function that takes as input

the no-purchase probabilities of the customer segments. We formalize this as follows.

The first-order conditions of Π (with respect to prices) can be written as

$$\left(p_k - \frac{1}{b_k}\right) \sum_{c \in M} w_c q_{c0}(p) s_{ck} - \sum_{j \in N} p_j e^{-p_j b_j} \sum_{c \in M} w_c q_{c0}(p)^2 s_{ck} s_{cj} = 0, \tag{2.23}$$

for all $k \in N$. We parameterize the no-purchase probabilities in (2.23) by defining, for all $k \in N$, the function $F_k : (0,1)^m \times \mathbb{R}^n \to \mathbb{R}$ as follows

$$F_k(x,p) := \left(p_k - \frac{1}{b_k}\right) \sum_{c \in M} w_c x_c s_{ck} - \sum_{j \in N} p_j e^{-p_j b_j} \sum_{c \in M} w_c x_c^2 s_{ck} s_{cj},$$

for all $x \in (0,1)^m$ and $p \in \mathbb{R}^n$. Correspondingly, define the mapping $F : (0,1)^m \times \mathbb{R}^n \to \mathbb{R}^n$ by $F(x,p) := [F_1(x,p), \ldots, F_n(x,p)]^\top$ for all $x \in (0,1)^m$ and $p \in \mathbb{R}^n$. The following proposition establishes that, for $x \in (0,1)^m$ fixed, a unique $p \in \mathbb{R}^n_{++}$ exists that solves $F(x,p) = 0$.

Proposition 2.7. *For all $x \in (0,1)^m$, there is a unique $p \in \mathbb{R}^n_{++}$ that solves $F(x,p) = 0$.*

Sketch of proof (detailed proof in Appendix 2.A.6). First, we show that, for $x \in (0,1)^m$, a $p \in \mathbb{R}^n_{++}$ exists that solves $F(x,p) = 0$ by relying on Brouwer's fixed point theorem. Then, for $x \in (0,1)^m$ fixed, we show that $F(x,\cdot)$ is injective by deriving that the Jacobian of F is a so-called P-matrix and by relying on Gale and Nikaido (1965). This shows that a unique $p \in \mathbb{R}^n_{++}$ exists that solves $F(x,p) = 0$ for $x \in (0,1)^m$ fixed. □

Remark 2.3. Note that Proposition 2.7 does not establish that, for $x, x' \in (0,1)^m$, $x \neq x'$, and $p, p' \in \mathbb{R}^n_{++}$ such that $F(x,p) = 0$ and $F(x',p') = 0$, it holds that $p \neq p'$.

With slight abuse of notation, Proposition 2.7 allows us to define $p : (0,1)^m \to \mathbb{R}^n_{++}$ as the unique positive solution to $F(x,p) = 0$ for $x \in (0,1)^m$, i.e., $p(\cdot)$ is such that $F(x,p(x)) = 0$ for all $x \in (0,1)^m$. Correspondingly, we define the revenue function $\pi' : (0,1)^m \to \mathbb{R}_{++}$ by

$$\pi'(x) := \Pi(p(x)), \text{ for } x \in (0,1)^m,$$

which leads us to the following corollary (which is the analogue of Equation (2.4)).

Corollary 2.2. *It holds that* $\max_{x \in [\underline{x}, \bar{x}]} \pi'(x) = \max_{p \in [\underline{p}, \bar{p}]} \Pi(p)$.

Proof. Let p^* be a maximizer of Π. By Proposition 2.1, p^* corresponds to a stationary point of Π, which implies that $F(q_0(p^*), p^*) = 0$. By Proposition 2.7, $\pi'(q_0(p^*)) =$

$\Pi(p^*)$. Since $p^* \in [\underline{p}, \bar{p}]$ (again, by Proposition 2.1), we have that $q_0(p^*) \in [\underline{x}, \bar{x}]$. Hence, $\max_{x \in [\underline{x}, \bar{x}]} \pi'(x) \geq \max_{p \in [\underline{p}, \bar{p}]} \Pi(p)$. This immediately implies the claimed equality, since a strict inequality leads to a contradiction (i.e., then p^* is not a maximizer). □

Clearly, a $(1-\epsilon)$-approximation to π' has a one-to-one correspondence with a $(1-\epsilon)$-approximation to Π. Namely, for an $x' \in [\underline{x}, \bar{x}]$ and $\epsilon > 0$ such that $\pi'(x') \geq (1-\epsilon) \max_{x \in [\underline{x}, \bar{x}]} \pi'(x)$, it holds that $\Pi(p(x')) \geq (1-\epsilon) \max_{p \in [\underline{p}, \bar{p}]} \Pi(p)$.

The apparent benefit of π' over π is that $\pi'(x)$ is well-defined for all $x \in (0,1)^m$, whereas $\pi(x)$ is only defined for $x \in X$ (which is a non-convex set in general, cf. Figure 2.1). Hence, by working with π' (instead of π) would alleviate the burden of having to check systematically, by means of the linear program $\texttt{LP}(x, \cdot)$, if $x \in X$ for $x \in [\underline{x}, \bar{x}]$. This potentially saves substantial computational effort (recall from the proof of Theorem 2.2 that the computational complexity of bounding the revenue over a single hypercube is dominated by the linear program that is used to check feasibility).

However, to construct a branch-and-bound algorithm (analogously to BNB) that uses π' to obtain a $(1-\epsilon)$-approximation for Π, we require a (local) Lipschitz constant of π'. It has proven to be very difficult to obtain a Lipschitz constant that is sufficiently small to make the algorithm practically useful. The most apparent approach is to use the implicit function theorem to bound the norm of $\frac{\partial \pi'(x)}{\partial x}$ over $[\underline{x}, \bar{x}]$ (or locally over $B(x, r)$ for some $x \in [\underline{x}, \bar{x}]$ and $r > 0$). This would mean bounding the norm of the following expression:

$$\begin{aligned} \frac{\partial \pi'(x)}{\partial x} &= \left(\frac{\partial p(x)}{\partial x} \right)^\top \cdot \left. \frac{\partial \Pi(p)}{\partial p} \right|_{p=p(x)} \\ &= \left(\left. \frac{\partial F(x,p)}{\partial p} \right|_{p=p(x)} \right)^{-1} \cdot \left. \frac{\partial F(x,p)}{\partial x} \right|_{p=p(x)} \cdot \left. \frac{\partial \Pi(p)}{\partial p} \right|_{p=p(x)} . \end{aligned} \tag{2.24}$$

Given that $p(x)$ in (2.24) is implicitly defined and given that there is no "easy" expression available for the matrix inverse in (2.24) makes it challenging—both globally and locally—to obtain sufficiently sharp bounds on the norm of (2.24). That being said, significant improvement in computation times are expected if a meaningful local Lipschitz constant for π' can be determined

For illustrative purposes, Figure 2.5 depicts the revenue function π' for two segments and twenty-five products. For this particular instance, parameters were sampled according to the simulation scheme form Section 2.4. Observe that the figure is a two-dimensional representation of the (originally) twenty-five-dimensional revenue function Π.

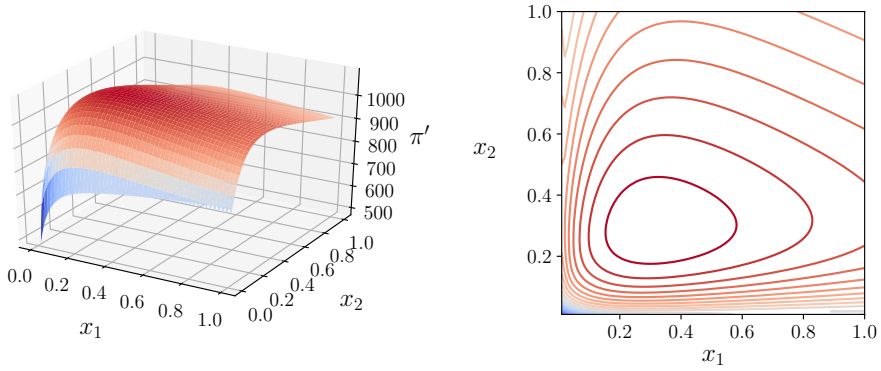


Figure 2.5: Illustration of the revenue function π' for $m = 2$ and $n = 25$. Parameters are sampled according to Section 2.4.

Finally, the following proposition alleviates the (potential) problem of having to solve a large system of non-linear equations to evaluate π'.

Proposition 2.8. *Let $x \in (0,1)^m$ and $r := rank(S)$ and assume, without loss of generality, that the first r columns of S are linearly independent. Then, the n-dimensional system of equations $F(x,p) = 0$ is equivalent to the following r-dimensional system:*

$$\begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_r \end{bmatrix} = \begin{bmatrix} b_1^{-1} \\ b_2^{-1} \\ \vdots \\ b_r^{-1} \end{bmatrix} + A_1^{-1} S_1^\top D S \begin{bmatrix} p_1 e^{-p_1 b_1} \\ \vdots \\ p_r e^{-p_r b_r} \\ \tilde{p}_1(p_1,\dots,p_r) e^{-\tilde{p}_1(p_1,\dots,p_r) b_{r+1}} \\ \vdots \\ \tilde{p}_{n-r}(p_1,\dots,p_r) e^{-\tilde{p}_{n-r}(p_1,\dots,p_r) b_n} \end{bmatrix}, \tag{2.25}$$

where $S_1 \in \mathbb{R}^{m\times r}$ and $S_2 \in \mathbb{R}^{m\times n-r}$ are such that $S = [S_1, S_2]$ and $\tilde{p} : \mathbb{R}^r \to \mathbb{R}^{n-r}$ is defined

by

$$\tilde{p}(p_1,\ldots,p_r) := \begin{bmatrix} b_{r+1}^{-1} \\ b_{r+2}^{-1} \\ \vdots \\ b_n^{-1} \end{bmatrix} + A_2^{-1} S_2^\top S_1 (S_1^\top S_1)^{-1} A_1 \left(\begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_r \end{bmatrix} - \begin{bmatrix} b_1^{-1} \\ b_2^{-1} \\ \vdots \\ b_r^{-1} \end{bmatrix} \right), \text{ where}$$

$$A_1 := \operatorname{diag}\left(\sum_{c\in M} w_c x_c s_{c1}, \ldots, \sum_{c\in M} w_c x_c s_{cr} \right),$$

$$A_2 := \operatorname{diag}\left(\sum_{c\in M} w_c x_c s_{c,r+1}, \ldots, \sum_{c\in M} w_c x_c s_{cn} \right), \text{ and } D := \operatorname{diag}\left(w_1 x_1^2, \ldots, w_m x_m^2 \right).$$

Sketch of proof (detailed proof in Appendix 2.A.6). Let $x \in (0,1)^m$. First, we show that $p_1(x)$, …, $p_r(x)$ is a solution to (2.25). Then, we show that $p_1(x), \ldots, p_r(x)$, $\tilde{p}_1(p_1(x),\ldots,p_r(x))$, …, $\tilde{p}_{n-r}(p_1(x),\ldots,p_r(x))$ is a solution to $F(x,p) = 0$. Since there exists a unique solution to $F(x,p) = 0$ by Proposition 2.7, the solution $p_1(x),\ldots,p_r(x)$ to (2.25) is also unique. Hence, there is a unique solution to (2.25) that corresponds with the unique solution to $F(x,p) = 0$. □

This proposition establishes that π' can be evaluated by solving an r-dimensional system of equations (instead of by solving the n-dimensional system $F(x,p) = 0$), where $r \leq \min(n,m)$ since $S \in \mathbb{R}^{m\times n}$. Given that the purpose of this paper is on pricing under the FML choice model when m (the number of segments) is relatively low, this potentially yields a substantial dimension reduction.

2.A Appendix: Proofs

2.A.1 Proofs of Section 2.3.2

Proof of Proposition 2.1

Let $c_1 > 0$, and define

$$\bar{a} := \max_{c\in M, j\in N} a_{cj} \text{ and } p_k^* := b_k^{-1} + \exp(\bar{a} - 1) \sum_{j\in N} b_j^{-1} + c_1,$$

for all $k \in N$, and $\mathcal{P} := \prod_{j\in N} \left[0, p_j^*\right]$. The proof consists of three steps. We first show that a global maximizer of Π exists in $\mathcal{P}$. Then, we show that global maxima only

occur in the interior of $\mathcal{P}$. This implies that all global maxima are attained at stationary points of Π on $\mathcal{P}$. Finally, we show that each stationary point, hence each maximizer, is contained in $[\underline{p}, \bar{p}]$.

Step 1. A global maximizer of the revenue function Π exists, and all maximizers are in $\mathcal{P}$.

Proof of Step 1. We show that for any price vector $p \in \mathbb{R}^n$, for which it holds that

$$p \notin \mathcal{P},$$

there exists a $p' \in \mathcal{P}$ such that $\Pi(p) < \Pi(p')$. To this end, observe that for all $j, k \in N$, $c \in M$, and $p \in \mathbb{R}^n$,

$$\frac{\partial q_{cj}(p)}{\partial p_k} = b_k q_{ck}(p) \left(q_{cj}(p) - 1_{\{j=k\}} \right),$$

and

$$\begin{aligned}
\nabla_k \Pi(p) &:= \frac{\partial \Pi(p)}{\partial p_k} \\
&= \sum_{c \in M} w_c \left\{ q_{ck}(p) + p_k \frac{\partial q_{ck}(p)}{\partial p_k} + \sum_{j \in N \setminus \{k\}} p_j \frac{\partial q_{cj}(p)}{\partial p_k} \right\} \\
&= \sum_{c \in M} w_c \left\{ q_{ck}(p) + p_k b_k q_{ck}(p) \left(q_{ck}(p) - 1 \right) + \sum_{j \in N \setminus \{k\}} p_j b_k q_{ck}(p) q_{cj}(p) \right\} \\
&= b_k \sum_{c \in M} w_c q_{ck}(p) \left\{ b_k^{-1} - p_k (1 - q_{ck}(p)) + \sum_{j \in N \setminus \{k\}} p_j q_{cj}(p) \right\} \\
&= b_k \sum_{c \in M} w_c q_{ck}(p) \left\{ b_k^{-1} - p_k + \sum_{j \in N} p_j q_{cj}(p) \right\} \\
&= b_k \sum_{c \in M} w_c q_{ck}(p) \left\{ b_k^{-1} - p_k \left(1 - \sum_{j \in N} q_{cj}(p) \right) + \sum_{j \in N} (p_j - p_k) q_{cj}(p) \right\}. \qquad (2.26)
\end{aligned}$$

From (2.26) it follows that negative prices are never optimal. To see this, let $p \in \mathbb{R}^n$ and suppose, by contradiction, and without loss of generality, that $p_n < 0$ and that $p_n \leq p_j$, for all $j \in N$. Then, from (2.26) it follows that

$$\nabla_n \Pi(p_1, p_2, \ldots, p_n) > 0.$$

Hence, there exists an $\epsilon > 0$ such that $\Pi(p_1, p_2, \ldots, p_n + \epsilon) > \Pi(p_1, p_2, \ldots, p_n)$ and, consequently, negative prices can never be optimal.

We proceed by showing that for any $p \in \mathbb{R}^n_+$ such that there is a $k \in N$ for which $p_k > p^*_k$, there is a $p' \in \mathcal{P}$ such that $\Pi(p') > \Pi(p)$. Let $p \in \mathbb{R}^n_+$ and suppose, without loss of generality, that $p_n > p^*_n$. Observe that, for all $c \in M$ and $j \in N$,

$$\begin{aligned} p_j q_{cj}(p) &\leq p_j \exp\left(a_{cj} - p_j b_j\right) \\ &\leq \max_{\tilde{p}_j \in \mathbb{R}} \tilde{p}_j \exp\left(a_{cj} - \tilde{p}_j b_j\right) \\ &= b_j^{-1} \exp\left(a_{cj} - 1\right) \\ &\leq b_j^{-1} \exp\left(\bar{a} - 1\right). \end{aligned} \tag{2.27}$$

It follows that

$$\begin{aligned} \nabla_n \Pi(p) &= b_n \sum_{c \in M} w_c q_{cn}(p) \left\{ b_n^{-1} - p_n + \sum_{j \in N} p_j q_{cj}(p) \right\} \\ &\overset{(a)}{\leq} b_n \sum_{c \in M} w_c q_{cn}(p) \left\{ b_n^{-1} - p_n + \exp\left(\bar{a} - 1\right) \sum_{j \in N} b_j^{-1} \right\} \\ &\overset{(b)}{<} b_n \sum_{c \in M} w_c q_{cn}(p) \left\{ b_n^{-1} - p^*_n + \exp\left(\bar{a} - 1\right) \sum_{j \in N} b_j^{-1} \right\} \\ &\overset{(c)}{=} -c_1 b_n \sum_{c \in M} w_c q_{cn}(p), \\ &< 0 \end{aligned} \tag{2.28}$$

where (a) follows from (2.27), (b) follows by the hypothesis that $p_n > p^*_n$, and (c) follows by definition of p^*_n. Hence, $\nabla_n \Pi(p_1, p_2, \ldots, \tilde{p}_n) < 0$ for all $\tilde{p}_n \in [p^*_n, p_n]$. By symmetry, for any $p \in \mathbb{R}^n_+$ such that there is a $k \in N$, for which it holds that $p_k > p^*_k$, we have that

$$\Pi(p_1, \ldots, p_k, \ldots, p_n) < \Pi(p_1, \ldots, p^*_k, \ldots, p_n).$$

Hence, for any $p' \notin \mathcal{P}$, there is a $p \in \mathcal{P}$ such that $\Pi(p) > \Pi(p')$. Since $\mathcal{P}$ is compact, and Π is continuous and bounded on $\mathcal{P}$, by the extreme value theorem a global maximizer of Π exists on $\mathcal{P}$. This completes the proof of Step 1.

Step 2. It holds that

$$\max_{p \in \prod_{j \in N} \left[0, p^*_j\right]} \Pi(p) = \max_{p \in \prod_{j \in N} \left(0, p^*_j\right)} \Pi(p).$$

Proof of Step 2. By contradiction, assume there is a p on the boundary of $\mathcal{P}$ that maximizes Π. This means that at least one of the following two cases hold:
Case (1). There is a $k \in N$ such that $p_k = 0$.

Case (2). There is a $k \in N$ such that $p_k = p_k^*$.
Assume that Case (1) holds. Let $p \in \mathcal{P}$ and, without loss of generality, let $k = n$. Then, since $p_n = 0$, and $p_j \geq 0$ for all $j \in N$, from (2.26) it follows that $\nabla_n \Pi(p) > 0$. Hence, there exists an $\epsilon > 0$ such that $p_n + \epsilon \in (0, p_n^*)$, and $\Pi(p_1, p_2, \ldots, p_n + \epsilon) > \Pi(p)$. In similar fashion, assume that Case (2) holds, that $p \in \mathcal{P}$ and, without loss of generality, that $p_n = p_n^*$. Then, from (2.26) it follows that $\nabla_n \Pi(p) < 0$, by similar arguments used to obtain (2.28). Hence, there exists an $\epsilon > 0$ such that $p_n - \epsilon \in (0, p_n^*)$, and $\Pi(p_1, p_2, \ldots, p_n - \epsilon) > \Pi(p)$. This completes the proof of Step 2.

Step 3. Each maximizer of Π is contained in $[\underline{p}, \bar{p}]$.

Proof of Step 3. By Step 1 and 2, a global maximizer of Π exists, and each global maximizer is in the interior of $\mathcal{P}$. This implies that all maximizers are attained at stationary points of Π. Let p be a maximizer of Π. From the first-order conditions, $\nabla \Pi(p) = 0$, it follows that, for all $k \in N$,

$$p_k = b_k^{-1} + \frac{\sum_{c \in M} w_c q_{ck}(p) \sum_{j \in N} p_j q_{cj}(p)}{\sum_{c \in M} w_c q_{ck}(p)}.$$

Since $p \in \mathcal{P}$, it holds that $p > 0$ and, therefore, $p_k \geq b_k^{-1} = \underline{p}_k$ for all $k \in N$. In addition,

$$\begin{aligned} p_k &= b_k^{-1} + \sum_{c \in M} \frac{w_c q_{ck}(p)}{\sum_{c' \in M} w_{c'} q_{c'k}(p)} \sum_{j \in N} p_j q_{cj}(p) \\ &\leq b_k^{-1} + \max_{c \in M} \max_{p \in \mathbb{R}_+^n} \sum_{j \in N} p_j q_{cj}(p) \\ &= \bar{p}_k, \end{aligned}$$

which completes the proof.

Remark 2.4. Based on the first-order conditions, Li and Webster (2017b) already observed that non-negative stationary points must lie in $[\underline{p}, \bar{p}]$. However, Li and Webster (2017b) do not show that such stationary points exist nor that a global maximizer exists.

Proof of Lemma 2.1

Let $x \in X$. Then, for all $p \in [\underline{p}, \bar{p}]$ such that $q_0(p) = x$, it holds that,

$$\begin{aligned}\Pi(p) &= \sum_{c \in M} w_c q_{c0}(p) \sum_{j \in N} p_j \exp\left(a_{cj} - p_j b_j\right) \\ &= \sum_{c \in M} w_c x_c \sum_{j \in N} s_{cj} p_j \exp\left(-p_j b_j\right) \\ &= \sum_{j \in N} b_j p_j \exp\left(-p_j b_j\right) k_j^\top x\end{aligned}$$

where we substitute x_c for $q_{c0}(p)$ and s_{cj} for $\exp\left(a_{cj}\right)$ to obtain the second equality, and substitute k_j for $b_j^{-1}\left[w_1 s_{1j}, \ldots, w_m s_{mj}\right]^\top$ for all $j \in N$ to obtain the final equality. We change decision variables from p_j to $z_j(p_j) = \exp\left(-p_j b_j\right)$ for all $j \in N$ to obtain the objective function $-\sum_{j \in N} z_j \log(z_j) k_j^\top x$. After the change of variables, it follows that the constraints $x_c = q_{c0}(p)$ are equal to $1 + \sum_{j \in N} s_{cj} z_j = \frac{1}{x_c}$, for all $c \in M$. The problem is concave since the function $-\log(y)y$ is concave on $y \in (0, e^{-1})$ and $z_j \in (0, e^{-1})$, since $0 < \underline{z}_j < \bar{z}_j < e^{-1}$ and $z_j \in [\underline{z}_j, \bar{z}_j]$, for all $j \in N$. Clearly, the constraints are linear.

Proof of Proposition 2.2

Let $x' \in X$, let $z' \in [\underline{z}, \bar{z}]$ be a maximizer of $\texttt{cnstr-rev}(x')$, and let $\epsilon > 0$. Suppose that

$$\pi(x') \geq (1 - \epsilon) \max_{x \in X} \pi(x).$$

For $p' = \left[-\frac{\log z_1'}{b_1}, \ldots, -\frac{\log z_n'}{b_n}\right]^\top$, observe that

$$p' \in [\underline{p}, \bar{p}],$$

since, for all $j \in N$, $p_j' = -\frac{\log z_j'}{b_j} \leq -\frac{\log \underline{z}_j}{b_j} = \bar{p}_j$, and $p_j' = -\frac{\log z_j'}{b_j} \geq -\frac{\log \bar{z}_j}{b_j} = \underline{p}_j$, given that $\underline{z}_j \leq z_j' \leq \bar{z}_j$, for all $j \in N$. It remains to show that $\Pi(p') \geq (1 - \epsilon) \max_{p \in [\underline{p}, \bar{p}]} \Pi(p)$. Observe that, for all $c \in M$,

$$\begin{aligned}q_{c0}(p') &= \frac{1}{1 + \sum_{j \in N} s_{cj} z_j'} \\ &= x_c', \end{aligned} \tag{2.29}$$

where the first equality holds by definition of p', and the second equality follows from the fact that $1 + \sum_{j \in N} s_{cj} z'_j = \frac{1}{x'_c}$, by virtue of z' being a solution to $\texttt{cnstr-rev}(x')$. Then,

$$\begin{aligned}
\Pi(p') &= \sum_{c \in M} w_c q_{c0}(p') \sum_{j \in N} p'_j \exp\left(a_{cj} - p'_j b_j\right) \\
&\overset{(a)}{=} \sum_{j \in N} p'_j b_j \exp\left(-p'_j b_j\right) k_j^\top x' \\
&\overset{(b)}{=} -\sum_{j \in N} z'_j \log z'_j k_j^\top x' \\
&\overset{(c)}{=} \max_{\substack{z \in [\underline{z}, \bar{z}]: \\ 1 + \sum_{j \in N} s_{cj} z_j = \frac{1}{x'_c}}} \sum_{j \in N} z_j \log z_j k_j^\top x' \\
&\overset{(d)}{=} \pi(x') \\
&\overset{(e)}{\geq} (1 - \epsilon) \max_{x \in X} \pi(x) \\
&\overset{(f)}{=} (1 - \epsilon) \max_{p \in [\underline{p}, \bar{p}]} \Pi(p).
\end{aligned}$$

To obtain (a), we rely on (2.29) to substitute $b_j k_j^\top x'$ for $\sum_{c \in M} w_c q_{c0}(p') s_{cj}$, for all $j \in N$ (recall that $\exp\left(a_{cj}\right) = s_{cj}$ and $k_j = b_j^{-1}[w_1 s_{1j}, \ldots, w_m s_{mj}]^\top$). Then, to obtain (b), we substitute $-b_j^{-1} \log(z'_j)$ for p'_j, for all $j \in N$. Equality (c) holds since z' is solution to $\texttt{cnstr-rev}(x')$, and (d) holds by definition of π. Finally, (e) holds by hypothesis, and (f) holds by Equation (2.4).

2.A.2 Proofs of Section 2.3.3

Proof of Proposition 2.3

By definition in (2.6), the dual equals

$$\mathcal{D}(x, \lambda, \underline{\nu}, \bar{\nu}) = \max_{z \in \mathbb{R}^n_{++}} \mathcal{L}(z, x, \lambda, \underline{\nu}, \bar{\nu}),$$

where $(\lambda, \underline{\nu}, \bar{\nu}) \in \mathbb{R}^m \times \mathbb{R}^n \times \mathbb{R}^n$, $x \in X$, and

$$\mathcal{L}(z, x, \lambda, \underline{\nu}, \bar{\nu}) = -\sum_{j \in N} z_j \log(z_j) k_j^\top x + \lambda^\top \left(Sz - \mu(x)\right) - \underline{\nu}^\top (z - \underline{z}) + \bar{\nu}^\top (z - \bar{z}).$$

Observe that $\mathcal{L}$ is concave in z over $\mathbb{R}_{++}$, since the function $-y\log(y)$ is concave on $\mathbb{R}_{++}$ and the linear part in the second term is clearly concave in z. Hence, it suffices to solve the first-order conditions of $\mathcal{L}$. To this end, observe that

$$\frac{\partial \mathcal{L}(z,x,\lambda,\underline{\nu},\bar{\nu})}{\partial z_j} = -(1+\log(z_j))k_j^\top x + \lambda^\top s_j - \underline{\nu}_j + \bar{\nu}_j$$

for all $j \in N$. It follows that the first-order conditions

$$-(1+\log(z_j))k_j^\top x + \lambda^\top s_j - \underline{\nu}_j + \bar{\nu}_j = 0,$$

are solved by

$$\tilde{z}_j(x,\lambda,\underline{\nu},\bar{\nu}) = \exp\left(\frac{\lambda^\top s_j + \bar{\nu}_j - \underline{\nu}_j}{k_j^\top x} - 1\right)$$

for all $j \in N$. This means that the dual equals

$$\begin{aligned}
\mathcal{D}(x,\lambda,\underline{\nu},\bar{\nu}) &= \max_{z\in\mathbb{R}^n_{++}} \mathcal{L}(z,x,\lambda,\underline{\nu},\bar{\nu}) \\
&= \mathcal{L}(\tilde{z}(x,\lambda,\underline{\nu},\bar{\nu}),x,\lambda,\underline{\nu},\bar{\nu}) \\
&= -\sum_{j\in N} \tilde{z}_j(x,\lambda,\underline{\nu},\bar{\nu}) \left(\frac{\sum_{c=1}^m \lambda_c s_{cj} + \bar{\nu}_j - \underline{\nu}_j}{k_j^\top x} - 1\right) k_j^\top x \\
&\quad + \sum_{c=1}^m \lambda_c \sum_{j\in N} s_{cj}\tilde{z}_j(x,\lambda,\underline{\nu},\bar{\nu}) \\
&\quad - \lambda^\top \mu(x) + \bar{\nu}^\top(\tilde{z}(x,\lambda,\underline{\nu},\bar{\nu}) - \bar{z}) - \underline{\nu}^\top(\tilde{z}(x,\lambda,\underline{\nu},\bar{\nu}) - \underline{z}) \\
&= \sum_{j\in N} \tilde{z}_j(x,\lambda,\underline{\nu},\bar{\nu})k_j^\top x - \lambda^\top \mu(x) + \underline{\nu}^\top \underline{z} - \bar{\nu}^\top \bar{z}
\end{aligned}$$

where $(\lambda,\underline{\nu},\bar{\nu}) \in \mathbb{R}^m \times \mathbb{R}^n \times \mathbb{R}^n$ and $x \in X$. The first equality follows by definition in (2.6). The second equality follows from the observation that $\mathcal{L}$ is concave in z, and that $\tilde{z}$ solves the first-order conditions. The third equality follows from substitution of $\tilde{z}$, and the final equality follows after rearranging terms. Strong duality holds by Slater's condition (Boyd and Vandenberghe, 2004, p. 226), since the objective function in `cnstr-rev`(x) is concave, and the constraints are affine.

It remains to show that, for $x \in X$, $\min_{(\lambda,\underline{\nu},\bar{\nu})\in\mathbb{R}^m\times\mathbb{R}^n_+\times\mathbb{R}^n_+} \mathcal{D}(x,\lambda,\underline{\nu},\bar{\nu})$ has a unique solution. To this end, we show that $\mathcal{D}$ is strictly convex in $(\lambda,\underline{\nu},\bar{\nu})$ over $\mathbb{R}^m \times \mathbb{R}^n_+ \times \mathbb{R}^n_+$ for $x \in X$ fixed. Let $x \in X$ and let $j \in N$. Observe that $\tilde{z}_j(x,\lambda,\underline{\nu},\bar{\nu})$ is strictly convex in

$(\lambda, \underline{\nu}, \bar{\nu})$, since it is the function composition of $\exp(\cdot)$, which is strictly convex, and

$$\frac{\lambda^\top s_j + \bar{\nu}_j - \underline{\nu}_j}{k_j^\top x} - 1,$$

which is an affine function $\mathbb{R}^{m+2} \to \mathbb{R}$. Since $k_j^\top x > 0$, it follows that $\tilde{z}_j(x, \lambda, \underline{\nu}, \bar{\nu}) k_j^\top x$ is strictly convex in $(\lambda, \underline{\nu}, \bar{\nu})$. This means that $\mathcal{D}$ is the sum of strictly convex functions (the summation in the first part of (2.7)) and convex functions (the linear functions in (2.7)), which implies that $\mathcal{D}$ is strictly convex in $(\lambda, \underline{\nu}, \bar{\nu})$. This completes the proof.

2.A.3 Proofs of Section 2.3.4

Proof of Lemma 2.2

Recall that, for $x \in (0,1)^m$ and $(\lambda, \underline{\nu}, \bar{\nu}) \in \mathbb{R}^m \times \mathbb{R}^n \times \mathbb{R}^n$, $\nabla \mathcal{D}(x, \lambda, \underline{\nu}, \bar{\nu})$ denotes the gradient of $\mathcal{D}$ with respect to the variable x, i.e., with respect to the first argument, evaluated at the point $(x, \lambda, \underline{\nu}, \bar{\nu})$. Correspondingly, for $c \in M$, we denote by $\nabla_c \mathcal{D}(x, \lambda, \underline{\nu}, \bar{\nu})$, the c^{th} element of $\nabla \mathcal{D}(x, \lambda, \underline{\nu}, \bar{\nu})$, i.e., the derivative of $\mathcal{D}$ with respect to x_c.

We first derive an expression for $\nabla_c \mathcal{D}$. Let $x \in (0,1)^m$ and $r > 0$ such that $B(x, r) \subset (0,1)^m$, and let $(\lambda, \underline{\nu}, \bar{\nu}) \in \mathbb{R}^m \times \mathbb{R}^n \times \mathbb{R}^n$. Then,

$$\begin{aligned}
\nabla_c \mathcal{D}(x, \lambda, \underline{\nu}, \bar{\nu}) &= \sum_{j \in N} \left(\frac{\partial k_j^\top x}{\partial x_c} \tilde{z}_j(x, \lambda, \underline{\nu}, \bar{\nu}) + k_j^\top x \frac{\partial \tilde{z}_j(x, \lambda, \underline{\nu}, \bar{\nu})}{\partial x_c} \right) + \frac{\lambda_c}{x_c^2} \\
&= \sum_{j \in N} \left(w_c \frac{s_{cj}}{b_j} \tilde{z}_j(x, \lambda, \underline{\nu}, \bar{\nu}) - w_c \frac{s_{cj}}{b_j} \tilde{z}_j(x, \lambda, \underline{\nu}, \bar{\nu}) \frac{\lambda^\top s_j + \bar{\nu}_j - \underline{\nu}_j}{k_j^\top x} \right) + \frac{\lambda_c}{x_c^2} \\
&= \frac{\lambda_c}{x_c^2} - w_c \sum_{j \in N} \frac{s_{cj}}{b_j} \tilde{z}_j(x, \lambda, \underline{\nu}, \bar{\nu}) \left(\frac{\lambda^\top s_j + \bar{\nu}_j - \underline{\nu}_j}{k_j^\top x} - 1 \right) \\
&= \frac{\lambda_c}{x_c^2} + w_c \sum_{j \in N} \frac{s_{cj}}{b_j} \phi(\tilde{z}_j(x, \lambda, \underline{\nu}, \bar{\nu})),
\end{aligned} \tag{2.30}$$

where we recall that $\phi(y) = -y \log(y)$, for $y \in \mathbb{R}_{++}$.

We bound the infinity norm of (2.30) from above as follows. Let $x \in X$ and $r > 0$ such that $B(x, r) \subset (0,1)^m$, and recall that $(\lambda^x, \underline{\nu}^x, \bar{\nu}^x)$ denotes the solution to the dual

problem (2.8). Then,

$$\begin{aligned}\max_{\tilde{x}\in B(x,r)} \|\nabla \mathcal{D}(\tilde{x},\lambda^x,\underline{\nu}^x,\bar{\nu}^x)\| &= \max_{\tilde{x}\in B(x,r),c\in\{1,\dots,m\}} \left| \frac{\lambda_c^x}{\tilde{x}_c^2} + w_c \sum_{j\in N} \frac{s_{cj}}{b_j} \phi(\tilde{z}_j(\tilde{x},\lambda^x,\underline{\nu}^x,\bar{\nu}^x)) \right| \\ &= \max_{\tilde{x}\in B(x,r),c\in\{1,\dots,m\}} \max \left\{ \frac{\lambda_c^x}{\tilde{x}_c^2} + w_c \sum_{j\in N} \frac{s_{cj}}{b_j} \phi(\tilde{z}_j(\tilde{x},\lambda^x,\underline{\nu}^x,\bar{\nu}^x)), \right. \\ &\quad \left. - \left(\frac{\lambda_c^x}{\tilde{x}_c^2} + w_c \sum_{j\in N} \frac{s_{cj}}{b_j} \phi(\tilde{z}_j(\tilde{x},\lambda^x,\underline{\nu}^x,\bar{\nu}^x)) \right) \right\}. \end{aligned} \tag{2.31}$$

Observe that, for all $\tilde{x} \in (0,1)^m$, $c \in M$, and $j \in N$,

$$\begin{aligned}\frac{\partial \phi(\tilde{z}_j(\tilde{x},\lambda^x,\underline{\nu}^x,\bar{\nu}^x))}{\partial \tilde{x}_c} &= \frac{\partial \phi(\tilde{z}_j(\tilde{x},\lambda^x,\underline{\nu}^x,\bar{\nu}^x))}{\partial \tilde{z}_j(\tilde{x},\lambda^x,\underline{\nu}^x,\bar{\nu}^x)} \frac{\partial \tilde{z}_j(\tilde{x},\lambda^x,\underline{\nu}^x,\bar{\nu}^x)}{\partial \tilde{x}_c} \\ = \underbrace{(1+\log(\tilde{z}_j(\tilde{x},\lambda^x,\underline{\nu}^x,\bar{\nu}^x)))}_{(a)} & \underbrace{w_c \frac{s_{cj}}{b_j} \tilde{z}_j(\tilde{x},\lambda^x,\underline{\nu}^x,\bar{\nu}^x) \left((\lambda^x)^\top s_j + \bar{\nu}_j^x - \underline{\nu}_j^x \right) \left(k_j^\top \tilde{x} \right)^{-2}}_{(b)}. \end{aligned} \tag{2.32}$$

For any maximizer z of $\texttt{cnstr-rev}(x)$, it holds that $z \in [\underline{z}, \bar{z}]$ by construction of $\texttt{cnstr-rev}(x)$. By strong duality (Proposition 2.3), we have that

$$\tilde{z}_1(x,\lambda^x,\underline{\nu}^x,\bar{\nu}^x), \dots, \tilde{z}_n(x,\lambda^x,\underline{\nu}^x,\bar{\nu}^x)$$

is a solution to $\texttt{cnstr-rev}(x)$. Hence, for all $j \in N$,

$$\tilde{z}_j(x,\lambda^x,\underline{\nu}^x,\bar{\nu}^x) = \exp\left(\frac{(\lambda^x)^\top s_j + \bar{\nu}_j^x - \underline{\nu}_j^x}{k_j^\top x} - 1 \right) \leq \bar{z}_j \leq \exp(-1),$$

and, since $k_j^\top x > 0$,

$$(\lambda^x)^\top s_j + \bar{\nu}_j^x - \underline{\nu}_j^x \leq 0. \tag{2.33}$$

By (2.33) and $k_j^\top \tilde{x} > 0$ for all $\tilde{x} \in (0,1)^m$, it follows that

$$\tilde{z}_j(\tilde{x},\lambda^x,\underline{\nu}^x,\bar{\nu}^x) = \exp\left(\frac{(\lambda^x)^\top s_j + \bar{\nu}_j^x - \underline{\nu}_j^x}{k_j^\top \tilde{x}} - 1 \right) \leq \exp(-1). \tag{2.34}$$

It follows that both (a) and (b) in (2.32) are non-positive and, therefore, (2.32) is non-

negative. Consequently, we have

$$\max_{\tilde{x}\in B(x,r)} \phi(\tilde{z}_j(\tilde{x}, \lambda^x, \underline{\nu}^x, \bar{\nu}^x)) = \phi(\tilde{z}_j(x + r, \lambda^x, \underline{\nu}^x, \bar{\nu}^x)), \text{ and}$$
$$\min_{\tilde{x}\in B(x,r)} \phi(\tilde{z}_j(\tilde{x}, \lambda^x, \underline{\nu}^x, \bar{\nu}^x)) = \phi(\tilde{z}_j(x - r, \lambda^x, \underline{\nu}^x, \bar{\nu}^x)).$$

Applying these equalities to (2.31) and observing that

$$\max_{\tilde{x}\in B(x,r)} \frac{\lambda_c^x}{\tilde{x}_c^2} \leq \frac{\lambda_c^x}{(x_c - \mathrm{sgn}(\lambda_c^x)\cdot r)^2}, \text{ and}$$
$$\min_{\tilde{x}\in B(x,r)} \frac{\lambda_c^x}{\tilde{x}_c^2} \leq \frac{\lambda_c^x}{(x_c + \mathrm{sgn}(\lambda_c^x)\cdot r)^2},$$

we obtain

$$\begin{aligned}\max_{\tilde{x}\in B(x,r)} \|\nabla \mathcal{D}(\tilde{x}, \lambda^x, \underline{\nu}^x, \bar{\nu}^x)\| \leq & \max_{c\in\{1,\ldots,m\}} \max \Bigg\{ \\ & \frac{\lambda_c^x}{(x_c - \mathrm{sgn}(\lambda_c^x)\cdot r)^2} + w_c \sum_{j\in N} \frac{s_{cj}}{b_j} \phi(\tilde{z}_j(x + r, \lambda^x, \underline{\nu}^x, \bar{\nu}^x)), \\ & \frac{-\lambda_c^x}{(x_c + \mathrm{sgn}(\lambda_c^x)\cdot r)^2} - w_c \sum_{j\in N} \frac{s_{cj}}{b_j} \phi(\tilde{z}_j(x - r, \lambda^x, \underline{\nu}^x, \bar{\nu}^x)) \Bigg\} \\ = & L(x, r).\end{aligned}$$

This completes the proof.

Proof of Proposition 2.4

Let $x \in X$ and $r > 0$ such that $B(x,r) \subset (0,1)^m$. First, suppose $L(x,r) \leq K(x,r)$. Then, it holds that

$$\begin{aligned}
\max_{\tilde{x} \in B(x,r) \cap X} \pi(\tilde{x}) &\overset{(a)}{=} \max_{\tilde{x} \in B(x,r) \cap X} \mathcal{D}\left(\tilde{x}, \lambda^{\tilde{x}}, \underline{\nu}^{\tilde{x}}, \bar{\nu}^{\tilde{x}}\right) \\
&\overset{(b)}{\leq} \max_{\tilde{x} \in B(x,r) \cap X} \mathcal{D}\left(\tilde{x}, \lambda^{x}, \underline{\nu}^{x}, \bar{\nu}^{x}\right) \\
&\leq \max_{\tilde{x} \in B(x,r)} \mathcal{D}\left(\tilde{x}, \lambda^{x}, \underline{\nu}^{x}, \bar{\nu}^{x}\right) \\
&\overset{(c)}{\leq} \mathcal{D}\left(x, \lambda^{x}, \underline{\nu}^{x}, \bar{\nu}^{x}\right) + \max_{\tilde{x} \in B(x,r)} \left\|\nabla \mathcal{D}\left(\tilde{x}, \lambda^{x}, \underline{\nu}^{x}, \bar{\nu}^{x}\right)\right\| \cdot \|\tilde{x} - x\| \\
&\overset{(d)}{\leq} \mathcal{D}\left(x, \lambda^{x}, \underline{\nu}^{x}, \bar{\nu}^{x}\right) + L(x,r) \cdot r \\
&\overset{(e)}{=} \pi(x) + L(x,r) \cdot r,
\end{aligned}$$

where (a) holds by strong duality (Proposition 2.3), (b) holds by weak duality, (c) holds by Lipschitz continuity of $\mathcal{D}$ in $\tilde{x}$ and convexity of $B(x,r)$, (d) holds by Lemma 2.2, and (e) holds by strong duality. This implies that the claim holds if $L(x,r) \leq K(x,r)$.

Now, suppose $K(x,r) \leq L(x,r)$. Let $\tilde{x} \in B(x,r) \cap X$. First, we show that the arc $\vec{x} : [0,1] \to \mathbb{R}^m$, defined by

$$\vec{x}(t) := \left[\frac{x_1 \tilde{x}_1}{t x_1 + (1-t)\tilde{x}_1}, \ldots, \frac{x_m \tilde{x}_m}{t x_m + (1-t)\tilde{x}_m}\right]^\top, \ t \in [0,1],$$

is completely contained in $B(x,r) \cap X$ (note that this arc connects x with $\tilde{x}$). Then, we show that $\pi(\vec{x}(t))$ is differentiable with respect to t for $t \in (0,1)$ and bound the absolute value of the derivative of π with respect to t over $(0,1)$. Finally, we use the mean-value theorem to obtain the claimed upper bound.

We now show that $\vec{x}(t) \in B(x,r) \cap X$ for all $t \in [0,1]$. First, we show that $\vec{x}(t) \in X$ for all $t \in [0,1]$. Observe that

$$\mu(\vec{x}(t)) = t\mu(\tilde{x}) + (1-t)\mu(x),$$

for $t \in [0,1]$. This implies that, for all $t \in [0,1]$, there exists a $z' \in [\underline{z}, \bar{z}]$ such that $Sz' = \mu(\vec{x}(t))$, namely $z' = t\tilde{z} + (1-t)z$ where $z, \tilde{z} \in [\underline{z}, \bar{z}]$ are such that $Sz = \mu(x)$ and $S\tilde{z} = \mu(\tilde{x})$. Hence, for all $t \in [0,1]$, there exists a $p' \in [\underline{p}, \bar{p}]$ such that $q_0(p') = \vec{x}(t)$,

namely $p' = \left[-\log(z_1')b_1^{-1}, \ldots, -\log(z_n')b_n^{-1}\right]^\top$, which shows that $\vec{x}(t) \in X$ for all $t \in [0,1]$. It also holds that $\vec{x}(t) \in B(x,r)$ for all $t \in [0,1]$ since, for all $c \in M$, $\vec{x}_c(0) = x_c$, $\vec{x}_c(1) = \tilde{x}_c$, and $\vec{x}_c(t)$ is strictly increasing (decreasing) in t if $x_c < \tilde{x}_c$ ($x_c > \tilde{x}_c$). This proves the claim that the arc $\vec{x}(t)$ for $t \in [0,1]$ is completely in $B(x,r) \cap X$.

We proceed by showing that $\pi(\vec{x}(t))$ is differentiable with respect to t, for all $t \in (0,1)$, and by bounding the absolute value of this derivative. Note that $\vec{x}(t)$ is differentiable with respect to t, for all $t \in (0,1)$, and therefore differentiability of $\pi(\vec{x}(t))$ with respect to t would follow by showing that π is differentiable in x. However, it may happen that $\vec{x}(t)$ lies on the boundary of the (closed) set X, in which case the derivative of $\pi(x)$ is not defined. To alleviate this, we extend the domain of π to $(0,1)^m$, as follows: for $x \in (0,1)^m$, let $\pi^e(x) := \min_{(\lambda,\underline{\nu},\bar{\nu}) \in \mathbb{R}^m \times \mathbb{R}^n_+ \times \mathbb{R}^n_+} \sum_{j \in N} \tilde{z}_j(x,\lambda,\underline{\nu},\bar{\nu}) k_j^\top x - \lambda^\top \mu(x) + \underline{\nu}^\top \underline{z} - \bar{\nu}^\top \bar{z}$, where $\tilde{z}_j(x,\lambda,\underline{\nu},\bar{\nu})$ is as in (2.5), defined for all $(x,\lambda,\underline{\nu},\bar{\nu}) \in (0,1)^m \times \mathbb{R}^m \times \mathbb{R}^n \times \mathbb{R}^n$. Similarly as in the proof of Proposition 2.3, one can show that $\tilde{z}_j(x,\lambda,\underline{\nu},\bar{\nu})$ is strictly convex in $(\lambda,\underline{\nu},\bar{\nu})$, for each $x \in (0,1)^m$. As a result, $\pi^e(x)$ exists and is well-defined, for each $x \in (0,1)^m$. In addition, π^e is a genuine extension of π, since Proposition 2.3 implies $\pi(x) = \pi^e(x)$ for all $x \in X$.

By the envelope theorem, $\pi^e(x)$ is differentiable with respect to x, for all $x \in (0,1)^m$. It follows that $\pi^e(\vec{x}(t))$ is differentiable with respect to t, for all $t \in (0,1)$. Since $\vec{x}(t) \in X$ for all $t \in (0,1)$, it follows that $\pi(\vec{x}(t))$ is differentiable with respect to t, for all $t \in (0,1)$.

We proceed by constructing an upper bound on the absolute value of this derivative. Recall that $\underline{\sigma}$ is the smallest singular value of S and define $h : (0,1)^m \times (0,1) \times \mathbb{R} \to \mathbb{R}$ by

$$h(x,r,\epsilon) := \frac{rn^{1/2}|\epsilon|}{\underline{\sigma} \min_{c \in M} x_c(x_c - r)}.$$

Let $t' \in (0,1)$ and $\epsilon \in \mathbb{R}$ such that $t' - \epsilon > 0$, $t' + \epsilon < 1$ and $h(x,r,\epsilon) < \min_{j \in N} \underline{z}_j$ (such an ϵ exists since $\underline{z}$ is bounded away from zero). Since $\left|\frac{\partial \vec{x}_c(t)}{\partial t}\right| = \frac{x_c \tilde{x}_c |\tilde{x}_c - x_c|}{(t(x_c - \tilde{x}_c) + \tilde{x}_c)^2} \leq r \frac{x_c \vee \tilde{x}_c}{x_c \wedge \tilde{x}_c}$ for all $t \in (0,1)$ and $c \in M$, it follows by the mean value theorem that

$$|\vec{x}_c(t' + \epsilon) - \vec{x}_c(t')| \leq |\epsilon| r \frac{x_c \vee \tilde{x}_c}{x_c \wedge \tilde{x}_c} \leq |\epsilon| r \frac{x_c + r}{x_c - r}. \tag{2.35}$$

Define $Z(\check{x}) := \{z \in \mathbb{R}^n : Sz = \mu(\check{x})\}$ for all $\check{x} \in X$. By (2.35), it follows that

$$\begin{aligned}
\pi(\vec{x}(t+\epsilon)) &= \max_{z \in Z(\vec{x}(t'+\epsilon)) \cap [\underline{z}, \bar{z}]} -\sum_{j \in N} z_j \log(z_j) k_j^\top \vec{x}(t'+\epsilon) \\
&\leq \max_{z \in Z(\vec{x}(t'+\epsilon)) \cap [\underline{z}, \bar{z}]} -\sum_{j \in N} z_j \log(z_j) k_j^\top \vec{x}(t') + |\epsilon| r \sum_{j \in N} b_j^{-1} \sum_{c \in M} w_c s_{cj} \frac{x_c + r}{x_c - r}. \\
&\leq \underbrace{\max_{z \in Z(\vec{x}(t'+\epsilon)) \cap [\underline{z}, \bar{z}]} -\sum_{j \in N} z_j \log(z_j) k_j^\top \vec{x}(t')}_{(*)} + |\epsilon| r n \bar{s} \underline{b}^{-1} \max_{c \in M} \frac{x_c + r}{x_c - r}. \qquad (2.36)
\end{aligned}$$

We claim that, for each $z \in Z(\vec{x}(t'+\epsilon)) \cap [\underline{z}, \bar{z}]$, there exists a

$$z' \in Z(\vec{x}(t')) \cap [\underline{z} - h(x, r, \epsilon), \bar{z} + h(x, r, \epsilon)]$$

such that

$$\|z - z'\| \leq h(x, r, \epsilon). \qquad (2.37)$$

To see this, suppose $z \in Z(\vec{x}(t'+\epsilon)) \cap [\underline{z}, \bar{z}]$. Let S^+ be the Moore-Penrose inverse of S (Moore, 1920, Penrose, 1955) and let $u \in \mathbb{R}^n$ such that $z = S^+ \mu(\vec{x}(t'+\epsilon)) + (I_n - S^+ S)u$. Such a $u \in \mathbb{R}^n$ exists since, for all $\check{x} \in X$,

$$Z(\check{x}) = \{S^+ \mu(\check{x}) + (I_n - S^+ S)u : u \in \mathbb{R}^n\}, \qquad (2.38)$$

see James (1978). In addition, let $z' := S^+ \mu(\vec{x}(t')) + (I_n - S^+ S)u$. Using that $\|S^+\| \leq n^{1/2} \|S^+\|_2 = n^{1/2} \underline{\sigma}^{-1}$, we find that

$$\begin{aligned}
\|z - z'\| &= \|S^+(\mu(\vec{x}(t'+\epsilon)) - \mu(\vec{x}(t')))\| \\
&\leq n^{1/2} \underline{\sigma}^{-1} \|\mu(\vec{x}(t'+\epsilon)) - \mu(\vec{x}(t'))\| \\
&= n^{1/2} \underline{\sigma}^{-1} \max_{c \in M} \left| \frac{1}{\vec{x}_c(t'+\epsilon)} - \frac{1}{\vec{x}_c(t')} \right| \\
&\leq |\epsilon| n^{1/2} \underline{\sigma}^{-1} \max_{c \in M} \left| \frac{1}{x_c} - \frac{1}{\tilde{x}_c} \right| \\
&\leq h(x, r, \epsilon),
\end{aligned}$$

which shows that inequality (2.37) holds for z'. Furthermore, $z' \in Z(\vec{x}(t))$ by definition of z' and given the identity in Equation (2.38), and

$$z' \in [\underline{z} - h(x, r, \epsilon), \bar{z} + h(x, r, \epsilon)],$$

since $z \in [\underline{z}, \bar{z}]$ and given (2.37). This proves the claim that for each $z \in Z(\vec{x}(t'+\epsilon)) \cap [\underline{z}, \bar{z}]$, there exists a $z' \in Z(\vec{x}(t')) \cap [\underline{z} - h(x,r,\epsilon), \bar{z} + h(x,r,\epsilon)]$ such that (2.37) holds. Using this result, we find that

$$\begin{aligned}
(*) &\leq \max_{z \in Z(\vec{x}(t')) \cap [\underline{z}-h(x,r,\epsilon), \bar{z}+h(x,r,\epsilon)]} -\sum_{j \in N} z_j \log(z_j) k_j^\top \vec{x}(t') \\
&\leq \max_{z \in Z(\vec{x}(t')) \cap [\underline{z}, \bar{z}]} -\sum_{j \in N} z_j \log(z_j) k_j^\top \vec{x}(t') + \\
&\quad h(x,r,\epsilon) \max_{z \in Z(\vec{x}(t')) \cap [\underline{z}-h(x,r,\epsilon), \bar{z}+h(x,r,\epsilon)]} \left\| \frac{\partial \sum_{j \in N} z_j \log(z_j) k_j^\top \vec{x}(t')}{\partial z} \right\| \\
&\leq \pi(\vec{x}(t')) + h(x,r,\epsilon) \max_{j \in N} \left(-\left[1 + \log(\underline{z}_j - h(x,r,\epsilon))\right] k_j^\top \vec{x}(t') \right). \qquad (2.39)
\end{aligned}$$

In a similar fashion, we can derive

$$\begin{aligned}
\pi(\vec{x}(t+\epsilon)) &= \max_{z \in Z(\vec{x}(t'+\epsilon)) \cap [\underline{z}, \bar{z}]} -\sum_{j \in N} z_j \log(z_j) k_j^\top \vec{x}(t'+\epsilon) \\
&\overset{(a)}{\geq} \max_{z \in Z(\vec{x}(t'+\epsilon)) \cap [\underline{z}, \bar{z}]} -\sum_{j \in N} z_j \log(z_j) k_j^\top \vec{x}(t') \\
&\quad - |\epsilon| r n \bar{s} \underline{b}^{-1} \max_{c \in M} \frac{x_c + r}{x_c - r} \\
&\overset{(b)}{\geq} \max_{z \in Z(\vec{x}(t')) \cap [\underline{z}+h(x,r,\epsilon), \bar{z}-h(x,r,\epsilon)]} -\sum_{j \in N} z_j \log(z_j) k_j^\top \vec{x}(t') \\
&\quad - |\epsilon| r n \bar{s} \underline{b}^{-1} \max_{c \in M} \frac{x_c + r}{x_c - r} \\
&\overset{(c)}{\geq} \pi(\vec{x}(t)) - h(x,r,\epsilon) \max_{j \in N} \left(-\left[1 + \log(\underline{z}_j - h(x,r,\epsilon))\right] k_j^\top \vec{x}(t') \right) \\
&\quad - |\epsilon| r n \bar{s} \underline{b}^{-1} \max_{c \in M} \frac{x_c + r}{x_c - r}, \qquad (2.40)
\end{aligned}$$

where (a) follows by (2.35), (b) follows by observing that for each $z \in Z(\vec{x}(t')) \cap [\underline{z} + h(x,r,\epsilon), \bar{z} - h(x,r,\epsilon)]$, there exists a $z' \in Z(\vec{x}(t'+\epsilon)) \cap [\underline{z}, \bar{z}]$ such that (2.37) holds, and (c) follows analogously to (2.39). Combining (2.36), (2.39), and (2.40), we conclude that,

for each $t' \in (0,1)$, the derivative of $\pi(\vec{x}(t))$ in t' satisfies

$$\begin{aligned}\left|\frac{\partial \pi(\vec{x}(t'))}{\partial t}\right| &= \left|\lim_{\epsilon \to 0} \frac{\pi(\vec{x}(t'+\epsilon)) - \pi(\vec{x}(t'))}{\epsilon}\right| \\ &\leq r n \bar{s} \underline{b}^{-1} \max_{c \in M} \frac{x_c + r}{x_c - r} \\ &\quad + \lim_{\epsilon \to 0} \frac{h(x,r,\epsilon) \max_{j \in N} \left(- \left[1 + \log(z_j - h(x,r,\epsilon))\right] k_j^\top \vec{x}(t') \right)}{\epsilon} \\ &= r \left\{ n \bar{s} \underline{b}^{-1} \max_{c \in M} \frac{x_c + r}{x_c - r} + \frac{n^{1/2} \max_{j \in N} \left(- \left[1 + \log(z_j)\right] k_j^\top \vec{x}(t') \right)}{\underline{\sigma} \min_{c \in M} x_c (x_c - r)} \right\}. \end{aligned} \tag{2.41}$$

It holds that

$$\begin{aligned}\max_{j \in N} \left(- \left[1 + \log(z_j)\right] k_j^\top \vec{x}(t') \right) &= \max_{j \in N} (\bar{p}_j b_j - 1) b_j^{-1} \sum_{c \in M} w_c s_{cj} \vec{x}_c(t') \\ &\leq \bar{s} \underline{b}^{-1} \max_{c \in M} \max_{p \in \mathbb{R}^n} \sum_{i \in N} p_i q_{ci}(p). \end{aligned} \tag{2.42}$$

By Theorem 2 in Li and Huh (2011), for each $c \in M$, it follows that $\max_{p \in \mathbb{R}^n} p_i q_{ci}(p) = \tilde{\Pi}_c$, where $\tilde{\Pi}_c$ is the unique solution to the univariate fixed-point equation

$$\tilde{\Pi}_c = \sum_{j \in N} b_j^{-1} s_{cj} e^{-(b_j \tilde{\Pi}_c + 1)}.$$

By implicit differentiation it follows that $\tilde{\Pi}_c$ is strictly increasing in s_{cj} and strictly decreasing in b_j for all $c \in M$ and $j \in N$. Using this observation, it can be derived that, for each $c \in M$,

$$\max_{p \in \mathbb{R}^n} p_i q_{ci}(p) \leq \underline{b}^{-1} W\left(n \bar{s} e^{-1}\right), \tag{2.43}$$

where $\underline{b}^{-1} W\left(n \bar{s} e^{-1}\right)$ is the solution to $\tilde{\Pi}_c = n \underline{b}^{-1} \bar{s} e^{-(\bar{b} \tilde{\Pi}_c + 1)}$, and W is the Lambert function (i.e., the inverse of $y e^y$), which is uniquely defined here since $n \bar{s} e^{-1} > 0$. By (2.42) and (2.43) it follows that Equation (2.41) can be bounded from above by

$$r n \bar{s} \underline{b}^{-1} \left(\max_{c \in M} \frac{x_c + r}{x_c - r} + \frac{n^{-1/2} \underline{b}^{-1} W\left(n \bar{s}\right)}{\underline{\sigma} \min_{c \in M} x_c (x_c - r)} \right) = r K(x,r), \tag{2.44}$$

where we used that $W(n \bar{s} e^{-1}) < W(n \bar{s})$. This shows that $\pi(\vec{x}(t))$ is differentiable with respect to t, for all $t \in (0,1)$, and that the absolute value of the derivative is bounded by the expression in Equation (2.44). By the mean-value theorem, it follows that $\pi(\vec{x}(1)) \leq$

$\pi(\vec{x}(0)) + rK(x,r)$, which translates to $\pi(\tilde{x}) \leq \pi(x) + rK(x,r)$. This proves the claim for $K(x,r) \leq L(x,r)$, since $\tilde{x} \in B(x,r) \cap X$ was chosen arbitrarily.

2.A.4 Proofs of Section 2.3.5

Proof of Proposition 2.5

We prove that for $x \in (0,1)^m$ and $r > 0$ such that $B(x,r) \subset (0,1)^m$, it holds that $B(x,r) \cap X \neq \emptyset$ if and only if $\texttt{LP}(x,r)$ is feasible. Let $[\cdot]^+$ and $[\cdot]^-$ denote operators that return the element-wise positive and negative parts, respectively, of vectors, i.e., $[v]^+ = [\max(0,v_1),\dots,\max(0,v_n)]^\top$ and $[v]^- = [-\min(0,v_1),\dots,-\min(0,v_n)]^\top$ for $v \in \mathbb{R}^n$.

($\Rightarrow$) Let $x \in (0,1)^m$ and $r > 0$ such that $B(x,r) \subset (0,1)^m$. Suppose $B(x,r) \cap X \neq \emptyset$. Let $x' \in B(x,r) \cap X$ and let $z' \in [\underline{z}, \bar{z}]$ such that $Sz' = \mu(x')$. Set the decision variables of $\texttt{LP}(x,r)$ to $z = z'$, $\delta^+ = [\mu(x) - \mu(x')]^+$, $\delta^- = [\mu(x) - \mu(x')]^-$, and $\gamma = \|\delta^+ + \delta^-\|$. Then, the constraints (2.14)-(2.18) of $\texttt{LP}(x,r)$ are all satisfied:

1. Constraint (2.14): For all $c \in M$,

$$\begin{aligned}\sum_{j \in N} s_{cj} z'_j + \delta_c^+ - \delta_c^- &= \mu_c(x') + [\mu_c(x) - \mu_c(x')]^+ - [\mu_c(x) - \mu_c(x')]^- \\ &= \mu_c(x') + \mu_c(x) - \mu_c(x') \\ &= \frac{1 - x_c}{x_c}.\end{aligned}$$

2. Constraint (2.15): $\gamma = \|\delta^+ + \delta^-\| = \max_{c \in M} |\delta_c^+ + \delta_c^-| \geq \delta_c^+ + \delta_c^-$ for all $c \in M$.
3. Constaint (2.16): $\delta_c^+ = \left[\frac{1}{x_c} - \frac{1}{x'_c}\right]^+ \leq \frac{1}{x_c} - \frac{1}{x_c + r} = \mu_c(x) - \mu_c(x+r)$.
4. Constraint (2.17): $\delta_c^- = \left[\frac{1}{x_c} - \frac{1}{x'_c}\right]^- \leq \frac{1}{x_c - r} - \frac{1}{x_c} = \mu_c(x-r) - \mu_c(x)$.
5. Constraint (2.18) is satisfied by definition of z'.

Hence, $\texttt{LP}(x,r)$ is feasible.

($\Leftarrow$) Let $x \in (0,1)^m$ and $r > 0$ such that $B(x,r) \subset (0,1)^m$. Suppose $\texttt{LP}(x,r)$ is feasible, and let $(\gamma, z, \delta^+, \delta^-)$ be a solution to $\texttt{LP}(x,r)$. We show that

$$x^\delta := \left[\frac{x_1}{1 + x_1 \cdot \left(\delta_1^- - \delta_1^+\right)}, \dots, \frac{x_m}{1 + x_m \cdot \left(\delta_m^- - \delta_m^+\right)}\right]^\top \in B(x,r) \cap X,$$

which then implies the claim, namely that $B(x,r) \cap X \neq \emptyset$. It follows by construction of the constraints (2.16) and (2.17) in $\texttt{LP}(x,r)$ that

$$\mu(x) - \delta^+ + \delta^- \in [\mu(x+r), \mu(x-r)].$$

Since for each $c \in M$ it holds that $\mu_c(x) = \frac{1-x_c}{x_c}$ is monotonically decreasing in x_c on $\mathbb{R}_{++}$, it follows that

$$\left\{\mu(x') : x' \in B(x,r)\right\} = [\mu(x+r), \mu(x-r)],$$

and, hence, $\mu^{-1}(\mu(x) - \delta^+ - \delta^-) \in B(x,r)$. Since the inverse of μ_c equals $\mu_c^{-1}(u) = \frac{1}{1+u_c}$, we have that

$$\mu^{-1}\left(\mu(x) - \delta^+ + \delta^-\right) = \begin{bmatrix} \frac{1}{1+\frac{1-x_1}{x_1} - \delta_1^+ + \delta_1^-} \\ \vdots \\ \frac{1}{1+\frac{1-x_m}{x_m} - \delta_m^+ + \delta_m^-} \end{bmatrix} = \begin{bmatrix} \frac{x_1}{1+x_1 \cdot \left(\delta_1^- - \delta_1^+\right)} \\ \vdots \\ \frac{x_m}{1+x_m \cdot \left(\delta_m^- - \delta_m^+\right)} \end{bmatrix} = x^\delta. \tag{2.45}$$

Hence, it follows that $x^\delta \in B(x,r)$.

We now show that x^δ is also in X. Since $(\gamma, z, \delta^+, \delta^-)$ is a solution to $\texttt{LP}(x,r)$, it holds that $Sz + \delta^+ - \delta^- = \mu(x)$ and, hence, we have that

$$Sz = \mu(x) - \delta^+ + \delta^-.$$

Observe that, by (2.45), $\mu(x^\delta) = \mu(x) - \delta^+ + \delta^-$. Hence, there is a $z' \in [\underline{z}, \bar{z}]$, namely $z' = z$, that solves $Sz' = \mu(x^\delta)$, which implies that $x^\delta \in X$.

Proof of Corollary 2.1

Let $x \in (0,1)^m$ and $r > 0$ such that $B(x,r) \subset (0,1)^m$, and suppose that $B(x,r) \cap X \neq \emptyset$. Let $(\gamma, z, \delta^+, \delta^-)$ be a solution to $\texttt{LP}(x,r)$. Observe that such a solution exists since, by Proposition 2.5, $B(x,r) \cap X \neq \emptyset$ implies that $\texttt{LP}(x,r)$ is feasible. Then,

$$\chi(x, \delta^+, \delta^-) \in B(x,r) \cap X$$

is an immediate consequence of the proof of Proposition 2.5, where we show that if $\texttt{LP}(x,r)$ is feasible, then $\chi(x, \delta^+, \delta^-) \in B(x,r) \cap X$.

We now show that $\chi(x, \delta^+, \delta^-) = x$ if $x \in X$. Let $x \in X$ and $r > 0$ such that

$B(x,r) \subset (0,1)^m$. Observe that $x \in X$ implies there is a $z \in [\underline{z}, \bar{z}]$ such that $Sz = \mu(x)$, since

$$\begin{aligned}
X &= \left\{ q_0(p) \,:\, p \in [\underline{p}, \bar{p}] \right\} \\
&= \left\{ x \in (0,1)^m \,:\, \exists p \in [\underline{p}, \bar{p}] \text{ such that } q_0(p) = x \right\} \\
&= \left\{ x \in (0,1)^m \,:\, \exists p \in [\underline{p}, \bar{p}] \text{ such that } \sum_{j \in N} s_{cj} \exp\left(-p_j b_j\right) = \frac{1 - x_c}{x_c}, \forall c \in M \right\} \\
&= \left\{ x \in (0,1)^m \,:\, \exists z \in [\underline{z}, \bar{z}] \text{ such that } \sum_{j \in N} s_{cj} z_j = \mu_c(x), \forall c \in M \right\} \\
&= \{ x \in (0,1)^m \,:\, \exists z \in [\underline{z}, \bar{z}] \text{ such that } Sz = \mu(x) \}
\end{aligned}$$

Hence, for each solution to $\texttt{LP}(x,r)$ it holds that $(\gamma, z, \delta^+, \delta^-) = (0, z, 0, 0)$, where $z \in [\underline{z}, \bar{z}]$ such that $Sz = \mu(x)$. This implies that $\chi(x, \delta^+, \delta^-) = \chi(x, 0, 0) = x$.

Proof of Proposition 2.6

Let $x \in (0,1)^m$ and $r > 0$ such that $B(x,r) \subset (0,1)^m$ and $B(x,r) \cap X \neq \emptyset$. Let $(\gamma, z, \delta^+, \delta^-)$ be a solution to $\texttt{LP}(x,r)$ (by Proposition 2.5 a solution exists). The claim is that

$$\min_{x' \in B(x,r) \cap X} \|\mu(x) - \mu(x')\| = \|\mu(x) - \mu(\chi(x, \delta^+, \delta^-))\|.$$

First of all, from Corollary 2.1 it follows that $\chi(x, \delta^+, \delta^-) \in B(x,r) \cap X$. Thus, it remains to show that there does not exist an $x' \in B(x,r) \cap X$ such that $x' \neq \chi(x, \delta^+, \delta^-)$ and $\|\mu(x) - \mu(x')\| < \|\mu(x) - \mu(\chi(x, \delta^+, \delta^-))\|$. Observe that the constraint $\gamma \geq \delta_c^+ + \delta_c^-$ for all $c \in M$ in $\texttt{LP}(x,r)$ implies that

$$\gamma = \|\delta^+ + \delta^-\|, \tag{2.46}$$

since $\|\delta^+ + \delta^-\| = \max_{c \in M} \delta_c^+ + \delta_c^-$ and—by virtue of γ being a minimizer of $\texttt{LP}(x,r)$—there is at least one $c \in M$ such that the inequality constraint $\gamma \geq \delta_c^+ + \delta_c^-$ is tight. In addition, it holds that

$$\begin{aligned}
\|\mu(\chi(x, \delta^+, \delta^-)) - \mu(x)\| &\overset{(a)}{=} \|\delta^+ - \delta^-\| \\
&\overset{(b)}{=} \|\delta^+ + \delta^-\| \\
&\overset{(c)}{=} \gamma.
\end{aligned}$$

Equality (a) follows from the observation that $\chi(x, \delta^+, \delta^-) = \mu^{-1}(\mu(x) - \delta^+ + \delta^-)$. Equality (b) holds since $\delta_c^+ \cdot \delta_c^- = 0$ for all $c \in M$ such that $\gamma = \delta_c^+ + \delta_c^-$. To see this, let $c \in M$ such that $\gamma = \delta_c^+ + \delta_c^-$. Suppose that both $\delta_c^+ > 0$ and $\delta_c^- > 0$ and, without loss of generality, suppose that $\delta_c^+ > \delta_c^-$. Then, setting δ_c^+ to $\delta_c^+ - \delta_c^-$ and δ_c^- to 0 would yield an improved solution contradicting that γ is a minimizer of $\texttt{LP}(x, r)$. Finally, (c) follows from (2.46). Hence, $\texttt{LP}(x, r)$ minimizes $\|\mu(\chi(x, \delta^+, \delta^-)) - \mu(x)\|$.

Suppose, by contradiction, that there exists an $x' \in B(x, r) \cap X$ such that $x' \neq \chi(x, \delta^+, \delta^-)$ and $\|\mu(x') - \mu(x)\| < \gamma$. Then, by setting $\delta^+ = [\mu(x) - \mu(x')]^+$ and $\delta^- = [\mu(x) - \mu(x')]^-$, we obtain a solution $\gamma = \|\delta^+ + \delta^-\|$ that is strictly smaller than γ, contradicting that γ is a minimizer.

2.A.5 Proofs of Section 2.3.6

Proof of Theorem 2.1

Let $\epsilon > 0$ and let $k = 1, 2, 3, \ldots$ index the iterations of $\texttt{BNB}(\epsilon)$, where a new iteration starts each time the algorithm passes the while clause on line 5. Let $\underline{\pi}^{(k)}$, $\bar{\pi}^{(k)}$, $r^{(k)}$, $\mathcal{X}_1^{(k)}$, and $\mathcal{X}_2^{(k)}$ be the value of $\underline{\pi}$, $\bar{\pi}$, r, $\mathcal{X}_1$, and $\mathcal{X}_2$, respectively, at the beginning of the k^{th} iteration, for all $k \in \mathbb{N}$. For example, $r^{(1)} = \frac{1}{2} \max_{c \in M}(\bar{x}_c - \underline{x}_c)$.

We first show that $\texttt{BNB}(\epsilon)$ terminates in a finite number of steps. By line 5 of $\texttt{BNB}(\epsilon)$, it suffices to show that there is a $k' \in \mathbb{N}$ such that

$$(1 - \epsilon)\bar{\pi}^{(k')} \leq \underline{\pi}^{(k')}. \tag{2.47}$$

Let $k_0 := 1 + \lceil \log_2(1 + n\bar{s}) \rceil$. Let $k \in \mathbb{N}$, $k \geq k_0$. With slight abuse of notation, for all $x \in \mathcal{X}_1^{(k)}$ such that $B(x, r^{(k)}) \cap X \neq \emptyset$, let $\chi(x)$ and $r(x)$ be the input to x^δ and r^δ, respectively, on line 10 and 11 of $\texttt{BNB}(\epsilon)$. By construction of k_0, it holds that, for each $x \in \mathcal{X}_1^{(k)}$ such that $B(x, r^{(k)}) \cap X \neq \emptyset$, the if-statement on line 13 is satisfied, i.e., $B(\chi(x), r(x)) \subset (0, 1)^m$. To see this, first note that $\chi(x) \in X$ for all $x \in \mathcal{X}_1^{(k)}$ such that $B(x, r^{(k)}) \cap X \neq \emptyset$, since $\chi(x) \in B(x, r^{(k)}) \cap X$ by Corollary 2.1. Furthermore, it holds

that $r(x) < \chi_c(x)$ for all $c \in M$ and $x \in \mathcal{X}_1^{(k)}$ such that $B(x, r^{(k)}) \cap X \neq \emptyset$, since

$$\begin{aligned} r(x) &= r^{(k)} + \|x - \chi(x)\| \\ &\overset{(a)}{\leq} 2r^{(k)} \\ &\leq 2r^{(k_0)} \\ &\overset{(b)}{=} \frac{1}{2^{k_0-1}} \max_{c \in M} (\bar{x}_c - \underline{x}_c) \\ &\leq \frac{1}{1 + n\bar{s}} \\ &\overset{(c)}{\leq} \min_{c \in M} \underline{x}_c. \end{aligned} \tag{2.48}$$

Here, (a) holds since $\chi(x) \in B(x, r^{(k)})$ by Corollary 2.1, (b) follows by using the identity $r^{(k)} = \frac{1}{2^k} \max_{c \in M} (\bar{x}_c - \underline{x}_c)$ for $k \in \mathbb{N}$, and (c) by observing that $\underline{x}_c = q_{c0}(\underline{p}) \geq (1 + n\bar{s})^{-1}$. This shows that the if-statement on line 13 is satisfied for each $x \in \mathcal{X}_1^{(k)}$ such that $B(x, r^{(k)}) \cap X \neq \emptyset$. It then follows by line 15 of $\texttt{BNB}(\epsilon)$ that

$$\bar{\pi}^{(k+1)} = \max_{x \in \mathcal{X}_1^{(k)} \,:\, B(x, r^{(k)}) \cap X \neq \emptyset} \pi^u \left(\chi(x), r(x) \right). \tag{2.49}$$

We proceed by deriving an upper bound for (2.49). By definition of π^u, cf. (2.12), the right-hand side of (2.49) is equal to

$$\max_{x \in \mathcal{X}_1^{(k)} \,:\, B(x, r^{(k)}) \cap X \neq \emptyset} \left\{ \pi(\chi(x)) + r(x) \cdot \left(L(\chi(x), r(x)) \wedge K(\chi(x), r(x)) \right) \right\}. \tag{2.50}$$

It follows that

$$\begin{aligned} \bar{\pi}^{(k+1)} &\leq \max_{x \in \mathcal{X}_1^{(k)} \,:\, B(x, r^{(k)}) \cap X \neq \emptyset} \left\{ \pi(\chi(x)) + 2r^{(k)} K(\chi(x), r(x)) \right\} \\ &\leq \underline{\pi}^{(k+1)} + 2r^{(k)} \max_{x \in \mathcal{X}_1^{(k)} \,:\, B(x, r^{(k)}) \cap X \neq \emptyset} K(\chi(x), r(x)), \end{aligned} \tag{2.51}$$

by construction of $\underline{\pi}^{(k+1)}$ on line 12, and where we used $r(x) \leq 2r^{(k)}$ for all $x \in \mathcal{X}_1^{(k)}$ such that $B(x, r^{(k)}) \cap X \neq \emptyset$, cf. (2.48). Let

$$K' := n\bar{s}\underline{b}^{-1} \left(1 + \frac{4}{\min_{c \in M} \underline{x}_c} + \frac{n^{-1/2} \underline{b}^{-1} W\left(n\bar{s}\right)}{\underline{\sigma} \min_{c \in M} \underline{x}_c^2} \right)$$

and observe that

$$\max_{x\in\mathcal{X}_1^{(k)}\,:\,B(x,r^{(k)})\cap X\neq\emptyset} K(\chi(x),r(x)) \le n\bar{s}\underline{b}^{-1} \max_{x\in\mathcal{X}_1^{(k)}\,:\,B(x,r^{(k)})\cap X\neq\emptyset} \left\{ \max_{c\in M} \frac{\chi(x)_c + 2r^{(k)}}{\chi(x)_c - 2r^{(k)}} + \frac{n^{-1/2}\underline{b}^{-1}W(n\bar{s})}{\underline{\sigma}\min_{c\in M}\chi(x)_c(\chi(x)_c - 2r^{(k)})} \right\} \le K'. \tag{2.52}$$

Let

$$k' := \left\lceil 2 + \log_2 \frac{K'}{\epsilon\underline{\pi}^{(1)}} \right\rceil \vee k_0.$$

Then, by (2.51) and (2.52), we find that

$$\begin{aligned}(1-\epsilon)\bar{\pi}^{(k')} &\le (1-\epsilon)\left(\underline{\pi}^{(k')} + 2K'r^{(k'-1)}\right)\\ &\le \underline{\pi}^{(k')} - \epsilon\underline{\pi}^{(k')} + \frac{K'}{2^{k'-2}}\max_{c\in M}(\bar{x}_c - \underline{x}_c)\\ &\le \underline{\pi}^{(k')} - \epsilon\underline{\pi}^{(k')} + \epsilon\underline{\pi}^{(1)}\\ &\le \underline{\pi}^{(k')},\end{aligned}$$

using the identity $r^{(k)} = 2^{-k}\max_{c\in M}(\bar{x}_c - \underline{x}_c)$ for $k \in \mathbb{N}$, and since $\underline{\pi}^{(k')} \ge \underline{\pi}^{(1)}$. This shows that (2.47) holds for k' and, hence, that $\texttt{BNB}(\epsilon)$ terminates in finite steps.

We now show that $\texttt{BNB}(\epsilon)$ delivers a $p' \in \mathbb{R}^n$ such that $\Pi(p') \ge (1-\epsilon)\max_{p\in\mathbb{R}^n}\Pi(p)$. Let $k \in \mathbb{N}$ be the number of completed iterations before the algorithm terminates. We claim that the set

$$\bigcup_{x\in\mathcal{X}_1^{(k)}} B\left(x, r^{(k)}\right) \tag{2.53}$$

contains all maximizers of π over X. By contradiction, suppose there is a maximizer x' of π in X that is not in (2.53). Then, for some $\kappa \in \{1,2,\ldots,k\}$, there is an $\tilde{x} \in \mathcal{X}_1^{(\kappa)}$ such that $x' \in B(\tilde{x}, r^{(\kappa)})$, but $\tilde{x} \notin \mathcal{X}_2^{(\kappa+1)}$ (here, we write $\mathcal{X}_2^{(k+1)}$ for the final value of $\mathcal{X}_2$). This implies that either $B\left(\tilde{x}, r^{(\kappa)}\right) \cap X = \emptyset$ (line 8) or $B\left(\tilde{x}, r^{(\kappa)}\right) \cap X \neq \emptyset$ and $\pi^u(x^\delta, r^\delta) < \underline{\pi}$ on line 14. The former contradicts that $x' \in X$ and, by Proposition 2.4, the latter contradicts optimality of x'. This proves that (2.53) contains all maximizers of π over X.

Writing $\underline{\pi}^{(k+1)}$ and $\bar{\pi}^{(k+1)}$ for the value of $\underline{\pi}$ and $\bar{\pi}$, respectively, if the algorithm termi-

nates, it follows that

$$\begin{aligned}\pi(x^\star) &\overset{(a)}{=} \underline{\pi}^{(k+1)} \\ &\overset{(b)}{\geq} (1-\epsilon)\bar{\pi}^{(k+1)} \\ &\overset{(c)}{\geq} (1-\epsilon) \max_{x \in \bigcup_{x' \in \mathcal{X}_1^{(k)}} B\left(x', r^{(k)}\right) \cap X} \pi(x) \\ &\overset{(d)}{=} (1-\epsilon) \max_{x \in X} \pi(x),\end{aligned}$$

where (a) holds by construction of $x^\star$ on line 12 of $\texttt{BNB}(\epsilon)$, (b) holds by the fact that $\texttt{BNB}(\epsilon)$ terminated after k iterations, and (c) and (d) follow by the fact that (2.53) contains all maximizers of π over X. Let z be a solution to $\texttt{cnstr-rev}(x^\star)$, and let

$$p' := \left[-\frac{\log z_1'}{b_1}, \ldots, -\frac{\log z_n'}{b_n}\right]^\top.$$

Then, $\Pi(p') \geq (1-\epsilon)\max_{p \in \mathbb{R}^n} \Pi(p)$ by Proposition 2.2, which completes the proof.

Proof of Theorem 2.2

Let $\epsilon > 0$. In the proof of Theorem 2.1, we show that the number of required iterations before $\texttt{BNB}(\epsilon)$ converges is at most

$$k' := \left\lceil 2 + \log_2 \frac{K'}{\epsilon \underline{\pi}^{(1)}} \right\rceil \vee k_0,$$

where $k_0 := 1 + \lceil \log_2 (1 + n\bar{s}) \rceil$ and

$$K' := n\bar{s}\underline{b}^{-1}\left(1 + \frac{4}{\min_{c \in M} \underline{x}_c} + \frac{n^{-1/2}\underline{b}^{-1} W(n\bar{s})}{\underline{\sigma} \min_{c \in M} \underline{x}_c^2}\right).$$

Hence, the total number of hypercubes evaluated by $\texttt{BNB}(\epsilon)$ is at most $2^{k'm}$. We have that

$$\underline{\pi}^{(1)} = \Pi(\underline{p}) = e^{-1} \sum_{c \in M} w_c \sum_{j \in N} s_{cj} b_j^{-1} \geq e^{-1} \min_{c \in M, j \in N} s_{cj} b_j^{-1},$$

which shows that $\underline{\pi}^{(1)}$ is bounded away from zero by a constant independent of m and n. In addition, $W(n\bar{s})$ is $\mathcal{O}(n^{1/2})$ and $(\min_{c \in M} \underline{x}_c)^{-1} = 1 + e^{-1} \max_{c \in M} \sum_{j \in N} s_{cj} \leq 1 +$

$n\bar{s}$. It follows that K' is $\mathcal{O}(n^3)$ and, thus, that the total number of hypercubes evaluated by $\texttt{BNB}(\epsilon)$ is $\mathcal{O}(\epsilon^{-m})$.

The time complexity of evaluating a single hypercube, i.e., a single iteration over lines 9-18 of $\texttt{BNB}(\epsilon)$, is $\mathcal{O}(m^{5.5}n^{5.5})$. To see this, observe that, for each hypercube, we solve the linear program LP from Section 2.3.5 to check feasibility and we minimize a convex function to evaluate π (on line 12) (we neglect the other arithmetic operations, since these are of lower time complexity). By relying on interior-point methods, the linear program is solved in $\mathcal{O}((2m+n+1)^{3.5}I^2)$ time, where $2m+n+1$ is the number of decision variables and I is the input size of LP in bits (Karmarkar, 1984). Since the input size in bits of LP is $\mathcal{O}(mn)$, solving LP takes $\mathcal{O}(m^{5.5}n^{5.5})$ time. To evaluate π, we solve the convex minimization problem in (2.8), which takes quadratic time in the number of variables when using (quasi)-Newton methods (Nocedal and Wright, 1999, p.198), i.e., its time complexity is $\mathcal{O}(n^2+m^2)$. Hence, the time complexity of evaluating a single hypercube is $\mathcal{O}(m^{5.5}n^{5.5})$.

It follows that the time complexity of $\texttt{BNB}(\epsilon)$ is $\mathcal{O}\left(\epsilon^{-m}n^{5.5+3m}\right)$.

2.A.6 Proofs of Section 2.5

Proof of Proposition 2.7

Let $x \in (0,1)^m$. We show there is a unique $p \in \mathbb{R}^n_{++}$ that solves $F(x,p)=0$. Let

$$\mathcal{P} := \prod_{k\in N} b_k^{-1}\left[1, 1+\frac{\sum_{c\in M} w_c x_c^2 s_{ck}\sum_{j\in N} s_{cj}}{e\sum_{c\in M} w_c x_c s_{ck}}\right] \subset \mathbb{R}^n_{++},$$

and recall that

$$F_k(x,p) = \left(p_k - b_k^{-1}\right)\sum_{c\in M} w_c x_c s_{ck} - \sum_{j\in N} p_j e^{-p_j b_j}\sum_{c\in M} w_c x_c^2 s_{ck}s_{cj},$$

for all $k \in N$.

First, we show that any $p \in \mathbb{R}^n_{++}$ that solves $F(x,p)=0$ is in $\mathcal{P}$ and that such a $p \in \mathcal{P}$ exists. Then, we show that F is injective in its second argument over $\mathcal{P}$. This proves there is a unique $p \in \mathcal{P}$ that solves $F(x,p)=0$.

Define the mapping $H : \mathbb{R}^n_{++} \to \mathbb{R}^n$ by $H(p) := F(x,p)+p$ and observe that the system $F(x,p)=0$ is equivalent to the fixed-point problem $H(p)=p$. Given the fact that, for

all $k \in N$,

$$H_k(p) = b_k^{-1} + \frac{\sum_{j \in N} p_j e^{-p_j b_j} \sum_{c \in M} w_c x_c^2 s_{ck} s_{cj}}{\sum_{c \in M} w_c x_c s_{ck}}, \tag{2.54}$$

and that $\max_{p_k \in \mathbb{R}} p_k e^{-p_k b_k} = b_k^{-1} e^{-1}$, it holds that any $p \in \mathbb{R}^n_{++}$ that solves $H(p) = p$ (and hence $F(x, p) = 0$) must be in $\mathcal{P}$. By Brouwer's fixed point theorem, such a solution exists, since H maps $\mathcal{P}$ to itself (and $\mathcal{P}$ is compact and convex).

We now show that F is injective over $\mathcal{P}$, i.e., $F(x, p) \neq F(x, p')$ for all $p, p' \in \mathcal{P}$ such that $p \neq p'$. To this end, we put the following definitions and theorem in place.

Definition 2.1 (principal sub-matrix)**.** For an $n \times n$ matrix A and $K \subseteq \{1, \ldots, n\}$, $K \neq \emptyset$, the *principal sub-matrix* $A_{[K]}$ is the sub-matrix of A when we only keep the rows and columns with indices in K.

Definition 2.2 (principal minor)**.** For an $n \times n$ matrix A and $K \subseteq \{1, \ldots, n\}$, $K \neq \emptyset$, the determinant of $A_{[K]}$ is a *principal minor* of A.

Definition 2.3 (P-matrix)**.** A square matrix is a P-matrix if all its principal minors are positive.

Theorem 2.3 (Gale-Nikaido, 1965, Theorem 4)**.** *Let $F : \Omega \to \mathbb{R}^n$ be a differentiable mapping, where Ω is a closed rectangular region in $\mathbb{R}^n$. Then, F is injective if the Jacobian of F is a P-matrix everywhere in Ω.*

We now show that the Jacobian of F is a P-matrix everywhere in $\mathcal{P}$. Let $p' \in \mathcal{P}$ and denote by $J \in \mathbb{R}^{n \times n}$ the Jacobian of F evaluated at p', i.e., $J_{ij} = \frac{\partial F_i(x, p')}{\partial p_j}$ for all $i, j \in N$. Observe that

$$J_{ij} = 1_{\{i=j\}} \sum_{c \in M} w_c x_c s_{ci} + (p'_j b_j - 1) e^{-p'_j b_j} \sum_{c \in M} w_c x_c^2 s_{ci} s_{cj},$$

for all $i, j \in N$. Let $K \subseteq \{1, \ldots, n\}$, $K \neq \emptyset$, with cardinality $k := |K|$, and let $\ell_1, \ldots, \ell_k$ index the elements of K. Then, it holds that

$$J_{[K]} = A_1 + UU^\top A_2, \tag{2.55}$$

where $U \in \mathbb{R}^{k\times m}_{++}$ with $U_{ic} = \sqrt{w_c}x_c s_{ci}$ for all $i \in K$ and $c \in M$,

$$A := \operatorname{diag}\left(\sum_{c\in M} w_c x_c s_{c\ell_1}, \ldots, \sum_{c\in M} w_c x_c s_{c\ell_k}\right), \text{ and}$$
$$B := \operatorname{diag}\left((p'_{\ell_1}b_{\ell_1} - 1)e^{-p'_{\ell_1}b_{\ell_1}}, \ldots, (p'_{\ell_k}b_{\ell_k} - 1)e^{-p'_{\ell_k}b_{\ell_k}}\right).$$

Taking the determinant of Equation (2.55) and applying the matrix determinant lemma, we find that

$$\det\left(J_{[K]}\right) = \det(A)\det\left(I_k + U^\top BA^{-1}U\right),$$

which is strictly positive since $\det(A) > 0$ and $\det\left(I_k + U^\top BA^{-1}U\right) > 0$, given that $I_k + U^\top BA^{-1}U$ is positive definite, as

$$U^\top BA^{-1}U = \left(B^{1/2}A^{-1/2}U\right)^\top \left(B^{1/2}A^{-1/2}U\right).$$

This shows that all principal minors of J are positive, i.e., that J is a P-matrix everywhere in $\mathcal{P}$. Hence, F is injective over $\mathcal{P}$ by Theorem 2.3. This shows there is a unique $p \in \mathbb{R}^n_{++}$ that solves $F(x,p) = 0$.

Proof of Proposition 2.8

Let $x \in (0,1)^m$ and $\check{p} := p(x)$ and let r, A_1, A_2, D, S_1, S_2, and $\tilde{p}(\cdot)$ be as defined in the claim. We first show that $\check{p}_1, \ldots, \check{p}_r$ is a solution to (2.25). The first r equations of $F(x,\check{p}) = 0$ can be written as

$$\left[\check{p}_1, \ldots, \check{p}_r\right]^\top - \left[b_1^{-1}, \ldots, b_r^{-1}\right]^\top = A_1^{-1}S_1^\top DS\left[\check{p}_1 e^{-\check{p}_1 b_1}, \ldots, \check{p}_n e^{-\check{p}_n b_n}\right]^\top. \tag{2.56}$$

Since $S_2^\top S_1(S_1^\top S_1)^{-1}S_1^\top = S_2^\top$, premultiplying both sides of (2.56) by

$$A_2^{-1}S_2^\top S_1(S_1^\top S_1)^{-1}A_1,$$

it follows by definition of $\tilde{p}(\cdot)$ that

$$\begin{aligned}\tilde{p}(\check{p}_1, \ldots, \check{p}_r) - \left[b_{r+1}^{-1}, b_{r+2}^{-1}, \ldots, b_n^{-1}\right]^\top &= A_2^{-1}S_2^\top DS\left[\check{p}_1 e^{-\check{p}_1 b_1}, \ldots, \check{p}_n e^{-\check{p}_n b_n}\right]^\top \\ &= \left[\check{p}_{r+1}, \check{p}_{r+2}, \ldots, \check{p}_n\right]^\top - \left[b_{r+1}^{-1}, b_{r+2}^{-1}, \ldots, b_n^{-1}\right]^\top,\end{aligned} \tag{2.57}$$

which shows that

$$\tilde{p}_j(\check{p}_1, \ldots, \check{p}_r) = \check{p}_{j+r} \text{ for all } j \in \{1, \ldots, n-r\}. \tag{2.58}$$

By substituting the identity (2.58) into (2.56) for all $j \in \{1, \ldots, n-r\}$, we obtain (2.25). This shows that $\check{p}_1, \ldots, \check{p}_r$ is a solution to (2.25).

We now show that

$$\check{p}_1, \check{p}_2, \ldots, \check{p}_r, \tilde{p}_1(\check{p}_1, \ldots, \check{p}_r), \tilde{p}_2(\check{p}_1, \ldots, \check{p}_r), \ldots, \tilde{p}_{n-r}(\check{p}_1, \ldots, \check{p}_r) \tag{2.59}$$

is a solution to $F(x, p) = 0$. First, observe that (2.56) and (2.58) imply that (2.59) is a solution to the first r equations of $F(x, p) = 0$. Analogously, (2.57) and (2.58) imply that (2.59) is a solution to the last $n - r$ equations of $F(x, p) = 0$. This shows that (2.59) solves $F(x, p) = 0$. Furthermore, since there is a unique solution to $F(x, p) = 0$ by Proposition 2.7, it holds that the solution to (2.25) is also unique. Hence, there is a unique solution to (2.25) that corresponds with the unique solution to $F(x, p) = 0$, which completes the proof.

Chapter 3

Numerical Performance of Dynamic Pricing and Learning Algorithms: A Controlled Experiment

3.1 Introduction

3.1.1 Background and motivation

It is becoming increasingly common in today's online marketplaces that sellers' pricing decisions are determined by algorithms. The most striking example is arguably Amazon.com, which made more than 2.5 million price changes each day during 2013—a staggering figure that—most likely—has only increased ever since.[1] Even the price of the Bible—not the most obvious candidate for dynamic pricing—changes dozens of times each year,[2] which reveals that algorithmic pricing has gained a strong foothold in today's business practice. The complexities of optimally adjusting prices in response to competitors' prices, changing market circumstances, interactions between products in the seller's own portfolio, consumer reviews, incomplete information about consumers' behavior, and many more factors that affect demand and revenue are obviously huge. To address these complexities, a large stream of scientific literature has emerged that designs pricing algorithms and analyzes their performance. A particularly large research area has evolved around the question of *learning*: how should a seller price its products to optimize profit when the price-demand relation is unknown upfront, and therefore has to be learned from accumulating sales data?

[1] https://www.profitero.com/2013/12/profitero-reveals-that-amazon-com-makes-more-than-2-5-million-price-changes-every-day/, visited on December 12, 2017.

[2] https://camelcamelcamel.com/Holy-Bible-James-Version-Burgundy/product/0718015592, visited on December 12, 2017

In recent years, a large number of studies have appeared that address this question from a monopolist's perspective (see literature review below). These research efforts have led to an understanding of the structure of optimal pricing strategies in a monopolist setting, in particular into the question of how much effort a seller should put into price experiments in order to strike the right balance between 'exploration' (conducting price experiments in order to learn the price-demand relation) and 'exploitation' (utilizing statistical knowledge to maximize profit).

For pricing and learning in a competitive environment, the picture is rather different. It turns out to be very difficult to give a useful qualitative assessment of a pricing strategy, for the simple reason that its performance depends on the (unknown) pricing behavior of competitors. A particular strategy may work very well when used against simplistic pricing rules, but perform much worse against sophisticated algorithms. Even the right performance measure is not clear (can one, e.g., improve upon the full information Nash equilibrium?). Not only is theoretical understanding limited, there also does not appear to be an extensive numerical study that compares the practical performance of different algorithms.

Thus, there is a serious lack of understanding of the structure of well-performing pricing strategies with learning and competition, while at the same time understanding these pricing strategies is increasingly important from a practical viewpoint. This motivated the organizers of the INFORMS Revenue Management & Pricing Section Conference 2017 to organize a dynamic pricing contest, in order to get insights into the numerical performance of different pricing strategies in a competitive environment with incomplete information, and so to gain insight into the properties of well-performing pricing policies. The results of this contest are reported in this chapter.

3.1.2 Literature review

The literature on 'learning and earning' from a monopolist's perspective has gained much attention in recent years: see, e.g., Araman and Caldentey (2009), Besbes and Zeevi (2009), Farias and van Roy (2010), Harrison et al. (2012), Broder and Rusmevichientong (2012), Chen and Farias (2013), Cheung et al. (2017), den Boer and Zwart (2014), Keskin and Zeevi (2014), den Boer and Zwart (2015), and Johnson Ferreira et al. (2016). A recent review of these and related references is provided by den Boer (2015). The main takeaway from this strand of literature is the importance of having the 'right' amount of *price experimentation*.

The importance of incorporating competition into these learning-and-earning models,

and the potential detrimental effect of ignoring competition, has been demonstrated by Schinkel et al. (2002), Tuinstra (2004), Bischi et al. (2004, 2007), Isler and Imhof (2008), Cooper et al. (2015), and Anufriev et al. (2013), building forth on earlier work by Kirman (Kirman, 1975, 1983, 1995, Brousseau and Kirman, 1992).

Various approaches have been adopted to incorporate competition into learning-and-earning problems. Bertsimas and Perakis (2006) consider least-squares learning in an oligopoly with finite inventories and linear demand function, and propose an algorithm for estimation and pricing. Kwon et al. (2009), Li et al. (2010), and Chung et al. (2012) adopt the framework of differential variational inequalities to study a capacitated oligopoly, propose an algorithm to solve these equations, and estimate unknown parameters via Kalman filtering. Perakis and Sood (2006) (see also Friesz et al. (2012)) take a robust-optimization approach to the dynamic oligopoly pricing problem, and study Nash equilibrium policies. Fisher et al. (2017) conduct a field experiment with randomized prices to estimate a consumer-choice model that does not require competitor sales observations, design a best-response pricing strategy, and test it with a field experiment for a leading Chinese online retailer.

A sample from the extensive economics and econometrics literature that study asymptotic behavior of pricing strategies in competitive environments is Cyert and DeGroot (1970), Kirman (1975), Aghion et al. (1993), Mirman et al. (1993), Fishman and Gandal (1994), Harrington (1995), Bergemann and Valimaki (1996), Gallego (1998), Alepuz and Urbano (1999), Rassenti et al. (2000), Belleflamme and Bloch (2001), Keller and Rady (2003), and Dimitrova and Schlee (2003). These papers typically assume that a particular learning method is used by the competitors, and study whether the price process converges to a Nash equilibrium.

The computer science literature also proposes several pricing-and-learning algorithms, see, e.g., Greenwald and Kephart (1999), Dasgupta and Das (2000), Tesauro and Kephart (2002), Kutschinski et al. (2003), Könönen (2006), Jumadinova and Dasgupta (2008, 2010), and Ramezani et al. (2011). For a further discussion of these and other relevant papers, we refer to Section 6.2 of den Boer (2015).

Finally, several simulation platforms have been designed to assess the performance of pricing policies, see, e.g., DiMicco et al. (2003) or Boissier et al. (2017).

3.1.3 Contributions

This chapter presents the results of the Dynamic Pricing Challenge, held on the occasion of the 17th INFORMS Revenue Management and Pricing Section Conference on June 29-30, 2017, at the Centrum Wiskunde & Informatica, Amsterdam, The Netherlands. For this challenge, participants were invited to submit pricing and learning algorithms that would compete for revenue in a broad range of simulated market environments in both duopoly and oligopoly settings. The extensive simulations that we ran allow us to describe the numerical performance of various pricing and learning algorithms and provides insight into the performance and properties of several types of policies. Given that the participants submitted a wide variety of algorithms—such as bandit-type models, customer choice models, econometric regression models, machine learning models, and greedy ad-hoc approaches—we are able to relate the performance of a broad range of algorithms to different market structures.

Hence, this chapter offers a framework to analyze various paradigms from the field of pricing and learning with competition and allows us to consider market dynamics that are analytically intractable and can not be empirically analyzed due to practical complications. As such, this chapter presents the results of a controlled experiment that improve our understanding of pricing and learning with competition and helps to guide future research. Our most important findings are as follows:

- The relative performance of the pricing and learning algorithms that we consider varies substantially across different market dynamics. Some algorithms perform well in competitive environments, whereas others are better at exploiting monopolist-like environments. None of the considered algorithms is able to dominate all the others in all settings.

- The relative performance of the pricing and learning algorithms that we consider varies substantially across oligopoly and duopoly markets. For example, algorithms based on linear demand models perform very well in duopoly competitions, whilst performing poorly in oligopolies.

- The algorithms that generate most revenue are more reliant on price-sensitive customers, making them vulnerable to intensified competition. Other algorithms are more robust in the sense that they were able to generate revenue from various types of customers and attract more loyal customers.

- A greedy algorithm that follows the lowest-priced competitor in a tit-for-tat fashion proves very difficult to outperform.

- Ignoring competition is increasingly harmful when competition is more fierce, i.e., when the number of competitors in the market is large and/or price sensitivity of the customers is high.
- The amount of exploration needs careful consideration as too much exploration hurts performance significantly.

The organization of the rest of this chapter is as follows. In Section 3.2, we describe the experimental design of this study. In Section 3.3 and 3.4, we present and discuss the results, respectively. In Section 3.5, we conclude with managerial insights and practical implications of our study for the industry.

3.2 Experimental Design

3.2.1 Experimental setting

This experiment was designed to resemble a market in which the competitors all sell a single product to a group of heterogeneous customers. The competitors have no information a priori on either the demand mechanism or the behavior of the other competitors and are required to post a price before each (discrete) time period. Furthermore, it was assumed that competitors can monitor each other's prices, but only observe their own sales (we refer to sales as the number of items sold). This is true for many markets in reality, especially in online retailing, where retailers can monitor competitors' prices without much effort. Thus, the participants of this experiment were required to design an algorithm that would accept as input their own realized sales and the historical prices of all competitors and, subsequently, as output returns their price for the period to come. In addition, we assumed there are no inventory restrictions and, to give the participants some direction, the following domain knowledge was made available: "it seems unlikely that posting prices higher than 100 is optimal." For convenience and to prevent compatibility issues, all participants were required to submit their pricing policy in Python 2.7.x or 3.x and no restrictions on the use of libraries were put in place.

To evaluate the performance of all submitted policies we ran 5000 simulations, where a single simulation consists of two different settings of competitive market environments:

- Duopoly competition: all competitors compete in a round-robin setup, i.e., each competitor competes with all other competitors in one-vs-one contests.
- Oligopoly competition: all competitors compete simultaneously against each other.

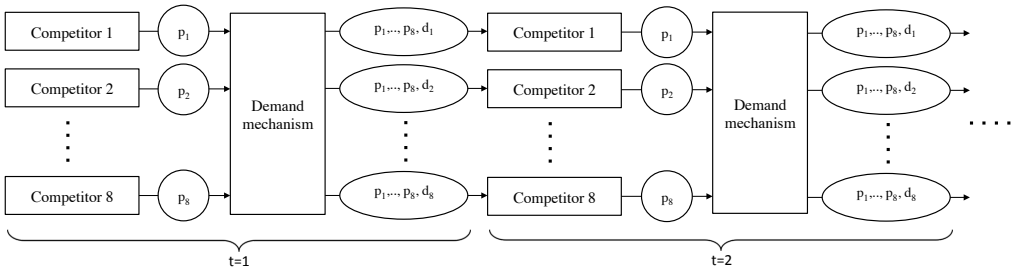


Figure 3.1: Overview of the first two time periods of the simulation of an oligopoly competition; p_k and d_k denote the price and sales quantity, respectively, of competitor $k \in \{1, \ldots, 8\}$.

In this experiment, there were eight competitors, which means that each simulation consists of $\binom{8}{2} = 28$ duopolies and one oligopoly. The oligopoly competition is especially insightful as it allows us to evaluate the competitors in a very competitive environment, whereas the duopoly competition can help us understand what the relative strength of the different competitors is.

In each of the 5000 simulations, the oligopoly competition and each of the duopoly competitions consists of 1000 discrete time periods. This means that for each $t \in \{1, \ldots, 1000\}$, each competitor posts a price for $t + 1$ and, subsequently, we generate sales quantities for period $t + 1$ from the undisclosed demand mechanism, and all competitors earn revenue accordingly. This iterative process is illustrated for the oligopoly competition in Figure 3.1 (for a duopoly competition, the scheme is the same except for the number of competitors, which is then two).

We measure the performance of all competitors and determine the winner of the pricing challenge as follows: In each simulation, we compute for each competitor its revenue share of that particular simulation by averaging

- the competitor's share of total revenue earned in the oligopoly competition and
- the competitor's share of total revenue earned in the duopoly competitions.

The latter is computed by aggregating all the revenue earned in the 28 duopoly competitions, and computing the competitors' revenue shares accordingly. More precisely, for simulation $i \in \{1, \ldots, 5000\}$, let x_{ij} be the revenue earned by competitor $j \in \{1, \ldots, 8\}$ in the oligopoly competition, and let y_{ijk} be the revenue earned by competitor $j \in \{1, \ldots, 8\}$ in the duopoly competition versus competitor $k \in \{1, \ldots, 8\}$. We define the

revenue share of competitor j in simulation i as follows:

$$\frac{1}{2}(\bar{x}_{ij} + \bar{y}_{ij}), \text{ where } \bar{x}_{ij} := \frac{x_{ij}}{\sum_{k=1}^{8} x_{ik}} \text{ and } \bar{y}_{ij} := \frac{\sum_{k=1}^{8} y_{ijk}}{\sum_{u=1}^{8}\sum_{k=1}^{8} y_{iuk}}. \tag{3.1}$$

Here, we set $y_{ijj} = 0$ for all $i \in \{1, \ldots, 5000\}$ and $j \in \{1, \ldots, 8\}$. In (3.1), $\bar{x}_{ij}$ and $\bar{y}_{ij}$ represent the oligopoly and duopoly revenue share, respectively, of competitor $j \in \{1, \ldots, 8\}$ in simulation $i \in \{1, \ldots, 5000\}$. Note that, as such, a competitor's own duopoly revenue share depends on the revenue earned in competitions between other competitors. This means it is not beneficial to earn a high revenue share in a duopoly competition where relatively little revenue is earned.

The final score for competitor $j \in \{1, \ldots, 8\}$ is simply the average over all its revenue shares, i.e., its final score equals $\frac{1}{5000}\sum_{i=1}^{5000}\frac{1}{2}(\bar{x}_{ij} + \bar{y}_{ij})$. This way of constructing a final score is a design choice of the pricing contest; clearly, numerous alternative ways to measure performance are conceivable.

3.2.2 Competitor algorithms

Table 3.1 summarizes the pricing and learning policies of all the competing algorithms. In the appendix contained in Section 3.A, we provide more detailed descriptions of the algorithms. The competitors rely on a wide range of demand models:

- linear models: OLS and WLS (ordinary and weighted least squares, respectively),
- bandit models: B-GRID, B-BUCKET, and B-MODEL (bandits where the arms pertain to price points in a grid, price buckets, and demand models, respectively),
- customer choice modeling: LOGIT,
- machine learning: ML, and
- greedy price-matching heuristic: GREEDY.

All competitors randomize prices in the first periods, and all competitors except LOGIT and GREEDY also engage in exploration later on, to capture possible non-stationary dynamics. Regarding the modeling of competitor behavior, most competitors use variants of exponential smoothing to predict competitor prices, and optimize own prices accordingly. Others model competitors' prices as multivariate normal random variables, use the median of historical prices as predictor, or ignore competition altogether. All non-bandit models use a line search to optimize their own revenue with respect to price,

except for WLS, which optimizes own revenue relative to the revenue earned by the competition.

Name	Demand Model	Pricing	Competitors' prices
LOGIT [1,226]	Finite mixture logit	During the first 100 periods the price is set as the minimum of the prices observed in the previous period. Then prices are optimized w.r.t. the demand model through a line search.	Multivariate normal distribution on historical prices
OLS [2,114]	Four linear regression models taking all combinations of price/log(price) and demand/log(demand). Select the model with the highest R^2.	During the first 40 periods and subsequently each time period with 5% probability a price is uniformly sampled from the interval $(0, 100)$. With probability 1% price is set to 0. Otherwise, prices are optimized through a line search.	Ignored
B-GRID [1,81]	ε-greedy multi-armed bandit algorithm with ten arms pertaining to the prices $10, 20, \ldots, 100$.	With probability $\varepsilon = 0.2$ an arm is chosen randomly, otherwise the arm with the highest average revenue so far is selected.	Ignored
B-BUCKET [2,160]	ε-greedy multi-armed bandit algorithm with ten arms pertaining to the price buckets $(0, 10], (20, 30], \ldots, (90, 100]$.	With probability 0.2 an arm is chosen randomly, otherwise the arm with the highest average revenue for the predicted competitor price (duopoly) or the mode of the competitors' prices (oligopoly) is chosen.	Exponential smoothing
B-MODEL [2,1010]	ε-greedy multi-armed bandit algorithm with four arms pertaining to three different demand models, each capturing different consumer behavior, and an epsilon-greedy algorithm in the fourth arm.	During the first 100 periods prices are set randomly, thereafter with probability 0.2 an arm is chosen randomly; otherwise offer the price suggested by the arm expected to generate the highest revenue, given a prediction of the competitors' prices.	Exponential smoothing
ML [1,325]	Least-squares, ridge, Lasso, Bayesian ridge, and stochastic gradient descent regression, and random forest (model selection through cross-validation).	Exploration cycles are initiated regularly and consist of 40 time periods in which prices are set according to a cosine function around the mean price level observed. Otherwise prices are optimized w.r.t. the demand model through a line search.	Exponential smoothing
GREEDY [1,11]	None	Price is set as minimum price observed in the previous period. If this price is lower than the 10% percentile of all the prices observed in the last 30 time periods, then the price is set as the maximum of this percentile and 5.	Ignored
WLS [1,387]	Weighted least squares where weights capture time-dependent aspects of demand.	Prices are set to maximize own revenue relative to competition. Prices are randomized during the first 10 periods prices and when own prices are constant for three subsequent periods.	Median of historical prices over variable window

Table 3.1: The variables in brackets after each competitor name corresponds to team size and lines of code, resp.

3.2.3 Demand mechanism

The design of our demand mechanism is built on the belief that it should resemble a competitive market with a heterogeneous customer base, as is often observed in practice. Meanwhile, we have to carefully manage the complexity of the demand mechanism to allow for evaluation, interpretation, and further analysis following the contest. To this end, we assume that the arrival process and demand mechanism are time-independent within a simulation. The customer population consists of three segments, namely loyal customers, shoppers, and scientists, who all have their own parameterized demand functions, as we describe in the following sections. We emphasize once more that the participants of this competition were not aware of any of the aforementioned assumptions regarding the market structure. All in all, we realize that the outcomes inevitably depend on the ground truth that we construct here, but it is intended to be versatile enough to reward the competitors that are best able to learn various types of demand dynamics.

Arrivals and population composition

In simulation $i \in \{1, \ldots, 5000\}$, we have the following arrival process and customer population composition. Customers arrive according to a Poisson process, with mean arrivals per time period equal to λ_i, where $\lambda_i \sim U(50, 150)$. Here and throughout, if we write $x \sim F$, we mean that x is sampled from (and not distributed as) F, i.e., x is a realization of F. Let θ_i^{sho}, θ_i^{loy}, and θ_i^{sci}, be the share of shoppers, loyals, and scientists, respectively, where $\theta_i^{\text{sho}} + \theta_i^{\text{loy}} + \theta_i^{\text{sci}} = 1$ and $\theta_i^{\text{sho}}, \theta_i^{\text{loy}}, \theta_i^{\text{sci}} \in (0, 1)$. The scientist segment consists of two subsegments, namely PhDs and professors, with respective shares of γ_i^{phd} and γ_i^{prof}, where $\gamma_i^{\text{phd}} + \gamma_i^{\text{prof}} = 1$ and $\gamma_i^{\text{phd}}, \gamma_i^{\text{prof}} \in (0, 1)$.

We summarize the sampling of arrivals in the code block "Arrival Process". Here, n is

Algorithm 2 Arrival Process

for $i \in \{1, \ldots, 5000\}$ **do**
  Sample arrival rate $\lambda_i \sim U(50, 150)$
  Sample segment shares $\theta_i^{\text{sho}}, \theta_i^{\text{loy}}, \theta_i^{\text{sci}}$
  Sample subsegment shares $\gamma_i^{\text{phd}}, \gamma_i^{\text{prof}}$
  for $t \in \{1, \ldots, 1000\}$ **do**
    Sample arrivals $n \sim \text{Poisson}(\lambda_i)$
    Sample segment arrivals $n^{\text{sho}}, n^{\text{loy}}, n^{\text{sci}} \sim \text{Multinom}(n, (\theta_i^{\text{sho}}, \theta_i^{\text{loy}}, \theta_i^{\text{sci}}))$
    Sample subsegment arrivals $n^{\text{phd}}, n^{\text{prof}} \sim \text{Multinom}(n^{\text{sci}}, (\gamma_i^{\text{phd}}, \gamma_i^{\text{prof}}))$

the number of arriving customers in a given period, and consists of n^{sho} shoppers, n^{loy} loyal customers, and n^{sci} scientists, i.e., $n = n^{\text{sho}} + n^{\text{loy}} + n^{\text{sci}}$. The scientist segment consists of n^{phd} PhDs and n^{prof} professors, i.e., $n^{\text{sci}} = n^{\text{phd}} + n^{\text{prof}}$.

Demand of shoppers and loyal customers

In simulation $i \in \{1, \ldots, 5000\}$, the willingness-to-pay (WTP) of shoppers is exponentially distributed with mean β_i^{sho}, where $\beta_i^{\text{sho}} \sim U(5, 15)$. This means that, in each time period in simulation i, we sample a WTP from the exponential distribution with mean β_i^{sho} for each arriving shopper. Each shopper whose WTP exceeds the lowest price offered in the market, buys from the competitor that offers the lowest price, and otherwise leaves without buying anything. Ties are broken randomly.

The WTP of the loyal customers in simulation $i \in \{1, \ldots, 5000\}$ is exponentially distributed as well, but with mean β_i^{loy}, where $\beta_i^{\text{loy}} = u \cdot \beta_i^{\text{sho}}$ and $u \sim U(1.5, 2.0)$. This makes the loyal customers relatively price insensitive compared to shoppers, as one would expect from loyal customers. In each time period, for each arriving loyal customer, we sample a WTP from the exponential distribution with mean β_i^{loy}, and assign this customer randomly to one of the competitors. This assignment determines to which competitor each customer is loyal to. We compare the WTP of each arriving loyal customer with the price set by the competitor that this customer is loyal to: they purchase if their WTP exceeds the price offered, and otherwise leave without making a purchase.

Demand scientists

The demand of the scientists is assumed to follow a finite-mixture logit model, or latent-class logit model, where the mixture comprises two components: professors and PhDs. This implies that the professors and PhDs both choose according to a logit model, but with their own respective parameters, which are set as follows.

Consider simulation $i \in \{1, \ldots, 5000\}$ and let m be the number of competitors in the market, i.e., $m = 2$ in any duopoly competitions and $m = 8$ in the oligopoly competitions. In a given period, suppose the posted prices in the market are equal to $p = (p_1, \ldots, p_m)$. We define the probability that an arriving PhD purchases from com-

petitor $k \in \{1, \ldots, m\}$ as follows:

$$q_{ki}^{\text{phd}}(p) = \frac{\exp\left(\alpha_i^{\text{phd}} - \beta_{mi}^{\text{phd}} \cdot p_k\right)}{1 + \sum_{j=1}^{n} \exp\left(\alpha_i^{\text{phd}} - \beta_{mi}^{\text{phd}} \cdot p_j\right)},$$

and we define $q_{ki}^{\text{prof}}(p)$ for professors in a similar fashion.

We sample the parameters α_i^{phd}, β_{mi}^{phd}, α_i^{prof}, and β_{mi}^{prof} to ensure that, if you optimize the price for each of the segments separately, then no unrealistically large differences in optimal prices occur. In doing so, we prevent that some customers are willing to pay a price that is orders of magnitudes larger than the price that other customers are willing to pay. To this end, we set α_i^{phd} equal to β_i^{sho}, and set β_{mi}^{phd} such that the optimal price for the PhDs is within 50% of the optimal price for the shoppers (which is equal to β_i^{sho}). This is achieved as follows,

$$\begin{aligned}
\alpha_i^{\text{phd}} &= \beta_i^{\text{sho}} \\
p_i^{\text{phd}} &:= \beta_i^{\text{sho}} \cdot u, \text{ where } u \sim U(0.50, 1.50) \\
\beta_{mi}^{\text{phd}} &= \frac{1 + W\left(m e^{\alpha_i^{\text{phd}} - 1}\right)}{p_i^{\text{phd}}}
\end{aligned}$$

where W is the Lambert function, i.e., $W(xe^x) = x$, which is uniquely defined in this case as $ne^{\alpha_i^{\text{phd}} - 1} > 0$ (Akcay et al., 2010). Here, p_i^{phd} is the price that maximizes revenue if the market consists solely of PhDs (see Theorem 6 in Akcay et al. (2010)):

$$\arg\max_{p \in \mathbb{R}_+^m} \sum_{k=1}^{m} p_k q_{ki}^{\text{phd}}(p) = \mathbb{1}_m p_i^{\text{phd}},$$

where $\mathbb{1}_m p_i^{\text{phd}}$ is understood to be an m-vector with each element equal to p_i^{phd}. By construction, p_i^{phd} is within 50% of the average WTP of shoppers. In similar fashion, we set

$$\begin{aligned}
\alpha_i^{\text{prof}} &= \alpha_i^{\text{phd}} \cdot u, \text{ where } u \sim U(1.00, 1.25) \\
p_i^{\text{prof}} &:= p_i^{\text{phd}} \cdot u, \text{ where } u \sim U(1.00, 1.50) \\
\beta_{mi}^{\text{prof}} &= \frac{1 + W\left(m e^{\alpha_i^{\text{prof}} - 1}\right)}{p_i^{\text{prof}}}
\end{aligned}$$

so that the optimal price in a market consisting of only professors would be higher than in a market that consists solely of PhDs, while keeping the price levels in line.

Demand mechanism summary

Summarizing, in each simulation $i \in \{1, \ldots, 5000\}$ and each time period $t \in \{1, \ldots, 5000\}$ of each duopoly and oligopoly, we see on average $\lambda_i \cdot \theta_i^{\text{sho}}$ arriving shoppers, $\lambda_i \cdot \theta_i^{\text{loy}}$ arriving loyals, $\lambda_i \cdot \theta_i^{\text{sci}} \cdot \gamma_i^{\text{phd}}$ arriving PhDs, and $\lambda_i \cdot \theta_i^{\text{sci}} \cdot \gamma_i^{\text{prof}}$ arriving professors. Each of these customer types chooses according to its own parameterized demand function (of which the parameters are constant throughout the simulation) as described in the previous sections. This implies that in some simulations $\theta_i^{\text{loy}} \approx 1$, meaning that all competitors are essentially monopolists, which should theoretically lead to higher prices, lower sales quantities, and relatively high revenues. Another extreme, when $\theta_i^{\text{sho}} \approx 1$, resembles a market in which there is perfect competition—each competitor offers the same product to arriving customers that purchase the cheapest alternative available. In this case, one would expect prices to spiral down over time.

The share of loyal customers in the market, which is $\frac{1}{3}$ on average, is independent of the number of competitors in the market. This means that in a duopoly, an arriving customer is a loyal customer that is loyal to a specific competitor with probability $\frac{1}{2} \cdot \frac{1}{3}$, whereas in the oligopoly, this probability equals $\frac{1}{8} \cdot \frac{1}{3}$. Therefore, by construction, we anticipate the market to be more competitive in the oligopoly setting than in the duopoly case, as one would expect from economic theory.

3.3 Results

In Section 3.3.1, we present a summary of the overall results. In Section 3.3.2 and 3.3.3, the results of the oligopoly and duopoly competitions, respectively, are considered.

3.3.1 Overall results

In Figure 3.2, we provide boxplots of the revenue shares for the oligopoly, the duopoly, and the overall competition. The boxplots in the left, middle, and right panel are based on $\bar{x}_{ij}$, $\bar{y}_{ij}$, and $\frac{1}{2}(\bar{x}_{ij} + \bar{y}_{ij})$, respectively (all defined in Equation (3.1)). Figure 3.2 reveals that LOGIT is the winner of the competition and that its success is primarily due to its superior performance in the oligopoly competitions (left panel). Overall, the differences

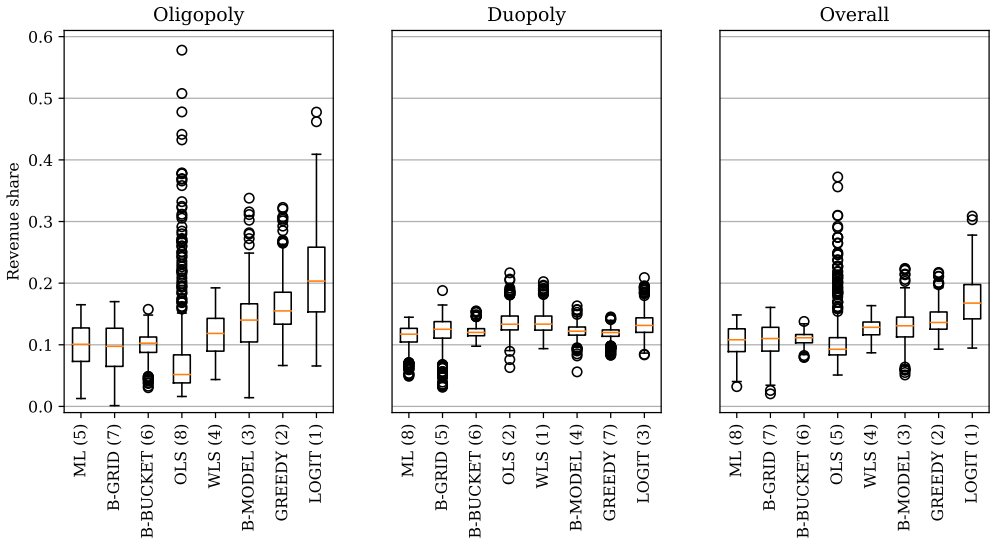


Figure 3.2: Boxplots of the revenue shares for the oligopoly competition, the duopoly competitions, and the overall competition. In parenthesis after the names of the algorithms are the rankings per competition part (e.g., WLS was the winner of the duopoly part). Only the first 500 of 5000 simulations are used to construct the boxplots for the sake of readability.

in performance in the oligopoly are substantial (left panel), whereas in the duopoly competitions (middle panel) it proves to be more difficult to outperform one another. Nevertheless, we observe remarkable differences in relative performance across the oligopoly and duopoly competitions. For example, OLS nearly earns the highest mean revenue share in the duopoly competitions (finishing second), while its performance in the oligopoly competition is on average the worst amongst all competitors. Note however that, despite its poor average, OLS did perform particularly well on some occasions (earning almost 60% of all revenue on one occasion). The other way around, we observe that competitor GREEDY performs poorly in the duopoly competitions, but that its performance in the oligopoly competition is relatively good, being second placed only after LOGIT. Furthermore, we observe that B-BUCKET, B-GRID, and ML are consistently outperformed, as they are in the bottom half in both the duopoly and oligopoly competition. In the following sections, we analyze the aforementioned observations in greater detail and make more detailed comparisons between the different pricing strategies in the market.

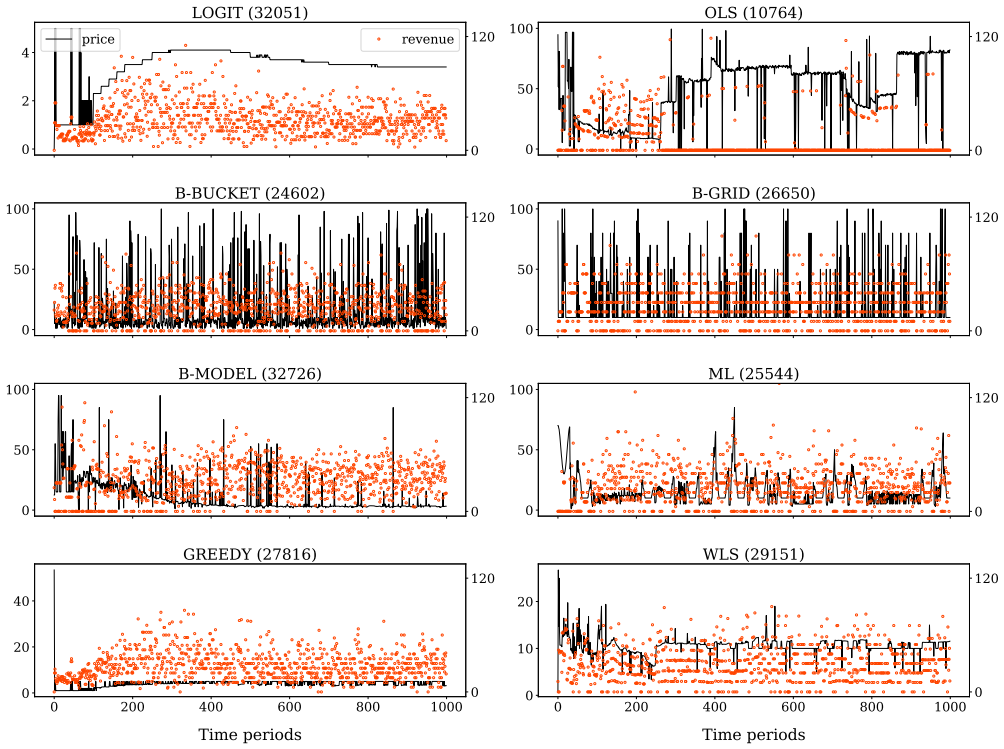


Figure 3.3: Realizations of prices and revenue for simulation 4955. The solid line represents the prices and pertains to the left axis. The red dots represent the revenue earned and pertain to the right axis. In parenthesis is the total accumulated revenue. The segment shares are equal to $\theta^{\text{sho}} = 0.05$, $\theta^{\text{loy}} = 0.72$, and $\theta^{\text{sci}} = 0.23$.

3.3.2 Oligopoly competition

In this section, we uncover what causes the substantial differences in performance in the oligopoly competition, which are observable in the first panel of Figure 3.2. We analyze how the competitors differ in terms of realized sales quantities and prices posted, and analyze how their performance varies as the market composition, i.e., the shares of segments, differs across simulations.

In Figure 3.3, we present the price and revenue realizations of a single simulation, to demonstrate what a simulation typically looks like. Although a single simulation is not representative for performance in general, we found that the simulation in Figure 3.3 is demonstrative for most of the scenarios that we visually inspected. In particular, the prices of LOGIT, B-MODEL, GREEDY, and WLS converge (after engaging in price exploration), whilst the other competitors show more erratic price paths. For example, B-

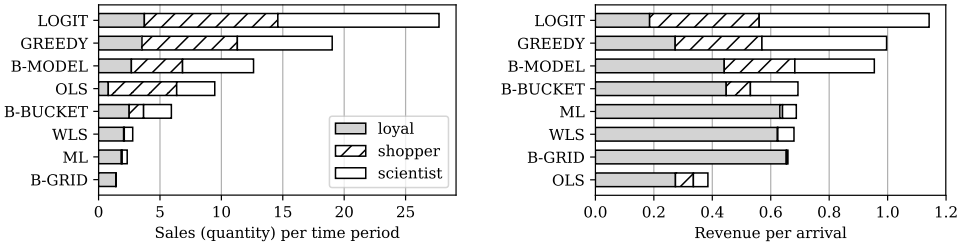


Figure 3.4: For the oligopoly competition (a) mean sales per time period and (b) mean revenue per arriving customer, split out over the three customer segments.

BUCKET and B-GRID put a lot of emphasis on exploration, without eventually converging to a competitive price range. This is arguably due to the fact that these algorithms engage in active price exploration with a 20% probability in each period. OLS initially seems to converge to a competitive price, but from around time period 300 onwards it fluctuates around a relatively high price level, earning hardly any revenue. We observe that ML initiates many exploration cycles around a cosine function, which affects its performance negatively in this stationary environment.

To show how the competitors differ in sales per customer segment, Figure 3.4(a) illustrates the mean sales per time period per segment for each competitor (note that sales is understood to be a quantity here and throughout). From this figure, it follows that the three best-performing algorithms in the oligopoly competitions, namely LOGIT, GREEDY, and B-MODEL (see the left panel of Figure 3.2), are also the ones that generate the highest sales. Furthermore, LOGIT not only generates the highest total sales, but it is also able to generate the highest sales per customer type. This is likely due to the fact that, if a competitor generates high sales in the shopper segment, this means that it is frequently the lowest priced competitor in the market, which means that the scientists and its loyal customers are also likely to buy. Thus, high sales in the shopper segment lead automatically to relatively high sales in the other two segments. In addition, it is remarkable that WLS generates much lower sales than OLS, while, according to Figure 3.2, WLS performs significantly better in terms of revenue generation. Arguably, this is due to the fact that WLS sells predominantly to the high-paying loyal segment, whereas OLS generates its revenue primarily from the shopper and scientist segments, which are more price sensitive. Overall, the competitors that attract the shoppers and scientists are capable of generating high sales, which concurs with the observation that for each competitor the potential sales from shoppers and scientists is much larger than from loyal customers (as was discussed in the final part of Section 3.2.3).

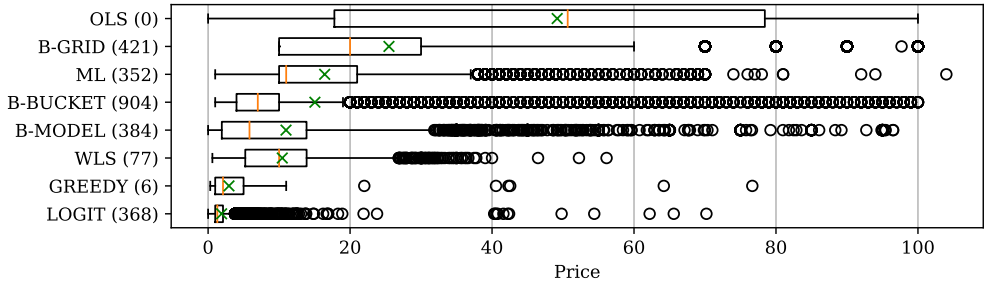


Figure 3.5: Boxplots of the prices posted in the oligopoly in 5000 randomly chosen time periods. The bars and crosses denote the median and mean, respectively. In brackets are the corresponding number of outliers.

Figure 3.4(b) illustrates the mean revenue per arriving customer for each competitor (e.g., LOGIT makes on average about 0.40 from an arriving shopper, whereas B-GRID practically earns nothing in this case). The figure reveals that the competitors' distributions of earnings over the customer segments vary substantially: the top performers (LOGIT, GREEDY, and B-MODEL) make the majority of their income from shoppers and scientists, while the other algorithms earn relatively much from the loyal customer segment. These observations concur with Figure 3.5, which contains boxplots of the prices posted by all competitors.

Regarding price experimentation, we observe from Figure 3.5 that OLS's policy induces a very wide price range, with its first quartile around twenty and the third quartile just below eighty, resembling a Gaussian distribution of prices. GREEDY does not engage in much price experimentation, as expected given its construction. The other boxplots indicate that the price distributions all have a heavy right tail; these competitors engage in exploration coincidentally, while pricing around a relatively small interval for most of the time.

Figure 3.6 illustrates the mean prices for different shares of loyal customers. This is insightful, since when the share of loyal customers increases from zero to one, the market moves from a very competitive market to a market in which every competitor is essentially a monopolist. One theoretically expects that competitors post higher prices once their pricing power increases. However, from the figure, we observe that the prices of three of the four worst-performers, namely OLS, B-GRID, and ML, do not increase in the share of loyal customers. On the other hand, the prices of LOGIT, GREEDY, WLS, B-BUCKET, and B-MODEL, do increase in the share of loyal customers. This indicates that these competitors are better capable of increasing prices when pricing power increases.

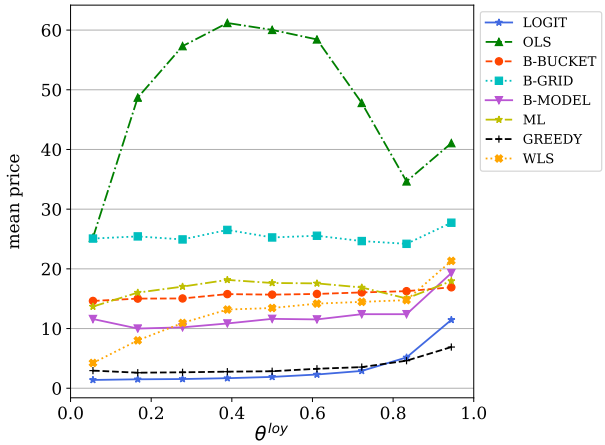


Figure 3.6: Mean prices for various levels of the share of loyal customers θ^{loy}.

Figure 3.7 and 3.8 illustrate the mean revenue per time period for various segment shares of loyal customers and scientists, respectively.[3] From Figure 3.7 we observe that, in general, revenues increase as the share of loyal customers increases, as one would theoretically expect. However, some competitors are better capable of exploiting the increase in pricing power than others—for example, WLS's relative performance improves substantially as θ^{loy} increases, while LOGIT's relative performance deteriorates. On the other hand, in Figure 3.8 we observe that performance across competitors diverges as the share of scientists increases. Most notably, LOGIT's relative performance increases substantially, which we attribute to the fact that LOGIT's demand specification is able to closely resemble the demand function of the scientists (which is a finite mixture of logit demand functions).

3.3.3 Duopoly competition

From the middle panel of Figure 3.2 it follows that, in the duopoly competitions, the differences in performance are smaller than in the oligopoly competition (left panel in Figure 3.2). There are two reasons why this is the case. First of all, the share of loyal customers is relatively large in the duopoly (see the discussion at the end of Section 3.2.3). This means that competitors have in general more pricing power, whilst only having to consider one other competitor, which makes it easier to earn revenue and more difficult to outperform one another. Second of all, the performance is not transitive in a sense

[3]The figure for the shopper segment is omitted to save space since it is very similar to Figure 3.7, except for that the revenues decrease in the shopper share.

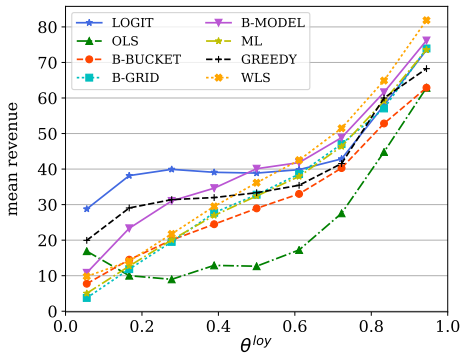


Figure 3.7: Mean revenue per time period for various levels of θ^{loy}.

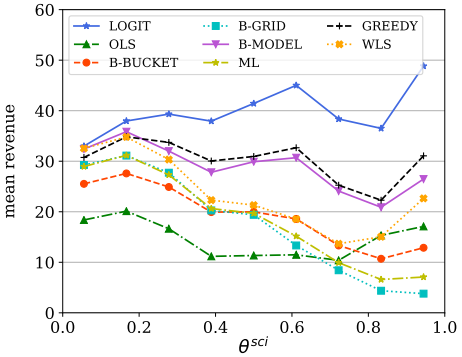


Figure 3.8: Mean revenue per time period for various levels of θ^{sci}.

that "if A beats B and B beats C, then A beats C", so that differences in performance tend to cancel out over the duopoly competitions. Nonetheless, the duopolies are interesting to analyze the relative performance of the various pricing policies.

For a single simulation, in Figure 3.9 we provide the realized prices and revenue of duopolies in which the top performer of the duopoly part, namely WLS, was involved.[4] The figure reveals that in the duopoly in which WLS and LOGIT compete, prices and revenue are both relatively low (this is not only the case in this example, but is a structural property, which we show below). It is interesting to see that GREEDY's performance is weakened by its mechanism that resets prices to 5 if prices get too low (see Table 3.1); these are the bumps that are visible in the bottom-right panel of Figure 3.9. As was the case in the oligopoly, OLS is not able to set a competitive price level consistently.

In Table 3.2, we provide the mean revenue per time period for the duopoly competitions. The last column contains the row-wise averages, which indicate how much revenue each algorithm makes on average. Similarly, the final row contains column-wise averages, which indicate how much revenue other algorithms were able to make against the corresponding algorithm (e.g., LOGIT makes on average 234 per time period, while the other competitors make on average 218 when competing with LOGIT). Table 3.2 confirms the earlier observation that the market between WLS and LOGIT is on average the smallest in terms of revenue ($153 + 96 = 249$ revenue per time period). In general, WLS proves to be very hard to generate revenue against since, on average, competitors earn only 175 per time period when competing with WLS. Meanwhile, WLS is able to generate substantial revenue with an average revenue per time period of 240. This is remarkable, as WLS performs poorly in the oligopoly.

[4] A plot of WLS vs B-GRID is omitted to save space, but is very similar to the plot of WLS vs B-BUCKET.

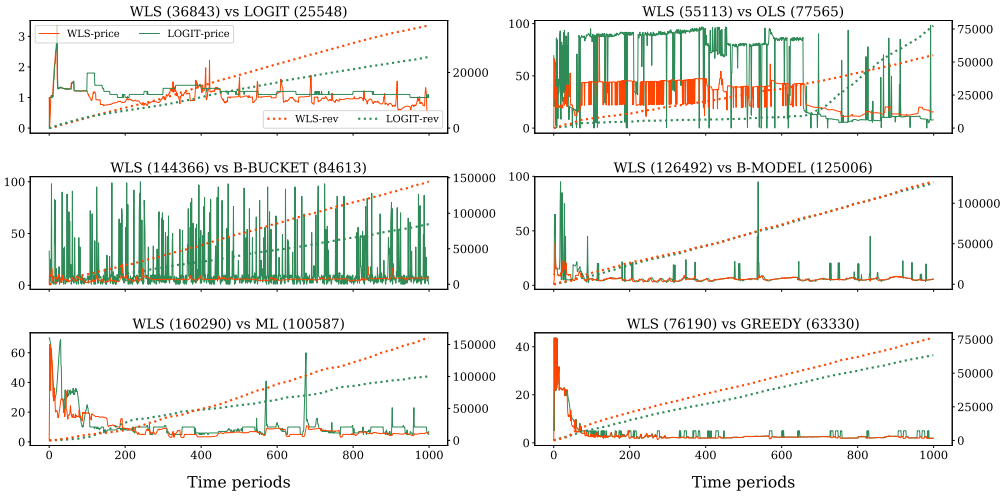


Figure 3.9: Realizations of prices and revenue for simulation 2285. The red and green solid lines represent the prices of WLS and the corresponding opponent, respectively, and pertain to the left axis. The red and green dotted lines represent the cumulative revenue earned by WLS and the corresponding opponent, respectively, and pertain to the right axis. In parenthesis is the total accumulated revenue. The segment shares are equal to $\theta^{\text{sho}} = 0.33$, $\theta^{\text{loy}} = 0.29$, and $\theta^{\text{sci}} = 0.38$.

	LOGIT	OLS	B-BUCKET	B-GRID	B-MODEL	ML	GREEDY	WLS	average
LOGIT		252	221	319	238	242	266	96	234
OLS	235		172	265	210	295	256	249	240
B-BUCKET	175	249		256	207	272	198	175	219
B-GRID	196	247	169		239	306	274	181	230
B-MODEL	250	231	185	241		270	176	218	224
ML	249	221	177	236	203		216	186	213
GREEDY	267	260	206	273	196	209		123	219
WLS	153	257	234	361	219	317	140		240
average	218	245	195	279	216	273	218	175	

Table 3.2: The pairwise mean revenue per time period for all duopolies. Each cell indicates how much revenue the algorithm in the corresponding row was able to earn against the algorithm in the corresponding column on average per time period. Green (red) indicates if the amount was higher (lower) than that of the corresponding opponent.

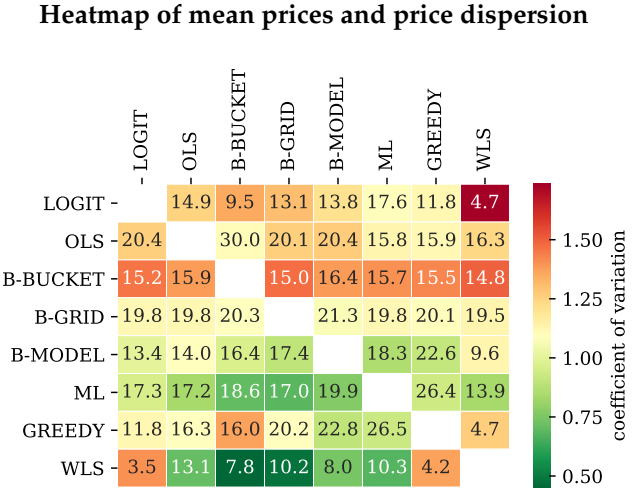


Figure 3.10: The number in each cell indicates the mean price that the algorithm in the corresponding row posted against the algorithm in the corresponding column. The color of each cell pertains to the coefficient of variation (i.e., the ratio of the standard deviation to the mean) and corresponds to the color bar on the right.

Table 3.2 also reveals that there is no competitor that loses against all other competitors. Even the worst performer (based on Figure 3.2), namely ML, is able to defeat other competitors, namely, LOGIT and GREEDY, which is remarkable as both perform very well overall. Based on Table 3.2, we observe that GREEDY has average performance—never earning much more (or much less) than its opponents. This is expected, since it simply follows its opponent's actions, without exploiting any weaknesses. The other top performer in the duopoly part, namely OLS, only beats B-GRID and ML according to Table 3.2. Nonetheless, we observe that OLS on average earns the same amount of revenue, namely 240, as WLS.

All in all, Table 3.2 reveals that each competitor's performance is very much dependent on the opponent's policy and that some algorithms that perform well in the oligopoly (e.g., GREEDY), struggle in the duopolies and vice versa (e.g., OLS). This, once more, confirms the intrinsic complexity of pricing and learning with competition.

To gain insight in the pricing levels of the competitors in the duopoly competitions, Figure 3.10 provides a heatmap of the mean prices and the coefficient of variation (the ratio of the standard deviation to the mean). The numerical values correspond to the mean prices posted and the color indicates the amount of dispersion in the prices. We observe that WLS consistently sets the lowest mean price and that OLS, which performed equally well according to Figure 3.2, set substantially higher prices than WLS. Thus,

OLS is able to be competitive while maintaining a higher overall price level. Other algorithms, e.g., B-GRID, also maintain a high mean price level, but are less successful in generating revenue. This can be explained by the fact that the price level of B-GRID is much less dependent on its competitor, i.e., its mean price posted is always around 20, whereas OLS varies its pricing level across its competitors, thereby being able to generate more revenue. Regarding the dispersion of prices, we observe that WLS, B-MODEL, and ML are the least experimental in setting prices and that especially B-BUCKET is very experimental, which is in correspondence with Figure 3.3 and 3.9.

3.4 Discussion

In terms of overall performance, LOGIT was the most effective algorithm in this competition — it earns most revenue in the oligopoly part and is also competitive in the duopoly part, where it finishes in third place, just behind OLS and WLS. Its success can partly be explained by the fact that its demand specification closely resembles the demand function of the scientist segment, which leads to the highest revenue per arriving scientist (see Figure 3.4(b) and 3.8). However, the primary reason for its success is undoubtedly the fact that LOGIT is able to identify and exploit the revenue potential of the shopper segment by setting low prices. This strategy negatively affects performance when pricing power increases (see Figure 3.7), but proves beneficial overall.

Arguably, if GREEDY would not have had arrangements in place to prevent a "race to the bottom", LOGIT's revenue would have been substantially lower. This indicates that LOGIT's low-pricing strategy does not perform well in all circumstances—its dependency on shoppers and scientists makes LOGIT vulnerable in case the market consists of more competitors that are aggressive on price. In terms of robustness, B-MODEL performs better as it is less reliant on the price-sensitive segments by selling more to loyal customers, whilst earning substantial revenue (Figure 3.4).

Regarding the two algorithms that rely on linear demand models, namely WLS and OLS, the difference in their performance between the duopoly and oligopoly parts is striking. In the duopoly setup, WLS and OLS are the two top performers, thereby confirming that the use of linear approximations for demand can give a simple and robust way to model the price-demand relationship. WLS is designed to maximize own revenue minus its competitor's revenue (see Table 3.1), which causes WLS to be very difficult to earn revenue against (see Table 3.2). On the other hand, OLS ignores competition altogether, which works surprisingly well in the duopoly competitions, and results in OLS posting much higher prices than WLS according to Figure 3.10 (but generating roughly the same

amount of revenue). In the oligopoly competitions, OLS is the worst-performing algorithm (Figure 3.2), which indicates that ignoring competition is increasingly harmful as competition increases (the relative performance of B-GRID, which also ignored competition, is also worse in the oligopoly competition than in the duopoly competition). The performance of WLS in the oligopoly is also poor, compared to its performance in the duopoly competitions. This shows that is ineffective to explicitly take into account all competitors' anticipated revenues.

Two of the bandit models, namely B-GRID and B-BUCKET, perform poorly due to a defect in their designs, which only allows prices in very crudely discretized price sets and prevents them to set low prices consistently, thereby greatly hindering their performance. The algorithms could be improved by making them more adaptive by, e.g., allowing poorly performing arms to be eliminated or adding additional arms close to the current optimal value to allow the algorithm to focus in on profitable prices.

The other bandit model, namely B-MODEL, is competitive and finishes third overall. It is designed to cope with different customer behaviors (e.g., bargain hunters and quality seekers, see the appendix in Section 3.A for details) by assigning a different demand model to each of its arms. The relative success of the B-MODEL appears to have been its ability to generate above-average revenues from each customer segment as, e.g., illustrated in Figure 3.4 (b). One of its pitfalls has been its high level of exploration of price points as illustrated in, e.g., Figures 3.3 and 3.5, especially in comparison to LOGIT and GREEDY.

The approach of ML heavily relies on machine learning methods to model the demand characteristics, and is designed to cope with non-stationarities, such as changes in the price elasticity over time or changes in the overall demand volume. In doing so, ML persists in engaging in exploration cycles, which hinders its performance in the stationary environment that we designed. This confirms the notion that one should only experiment if the anticipated long-term revenue of doing so outweighs the short-term cost.

Certainly the most simple strategy, namely GREEDY's ad-hoc approach of matching the lowest price in the market, turns out to be very effective in the oligopoly competition. Generally, it follows whoever set the lowest price, and is thereby able to generate substantial revenue—especially from scientists and shoppers. The arrangements that GREEDY put in place to prevent downward price spirals are initiated frequently (Figure 3.9), which could otherwise have led to even lower prices, and presumably deteriorating revenues. In the duopoly competitions, no competitor outperforms GREEDY significantly, however, GREEDY is also not able to exploit competitors' weaknesses ei-

ther, leading to average performance.

3.5 Conclusion and Managerial Insights

This chapter presented the results of the Dynamic Pricing Challenge, held on the occasion of the 17th INFORMS Revenue Management and Pricing Section Conference on June 29-30, 2017, at the Centrum Wiskunde & Informatica, Amsterdam, The Netherlands. The participants of this pricing challenge submitted a wide variety of pricing and learning algorithms, which we analyzed in a simulated market environment with competition. As such, this chapter presents a framework in which various paradigms from the field of pricing and learning with competition are analyzed by means of a controlled experiment. This allows us to consider market dynamics that are not analytically tractable or cannot be empirically analyzed due to practical complications.

Our analysis reveals a number of interesting insights, both from a practical and scientific point of view. First of all, we show that the relative performance of pricing and learning algorithms varies substantially across oligopoly and duopoly markets and across different market dynamics. This confirms the intrinsic complexity of pricing and learning in the presence of competition. Most notably, none of the considered algorithms is able to consistently outperform the other algorithms—each algorithm meets its Waterloo at some point in the competition. This reveals that algorithm design needs careful consideration, and that the structure and dynamics of the market need to be taken into account to determine which algorithm is the best fit. Second, a greedy algorithm that follows the lowest-priced competitor in a tit-for-tat fashion proves very difficult to outperform. Especially in oligopolistic markets, it is able to generate substantial revenue from price-sensitive customers, whilst showing average performance in the duopoly competitions. Third, although the eventual winner was determined by revenue earned, the results reveal that some algorithms are better capable of attracting customers from different segments, thereby being less reliant on one specific segment and, therefore, being more robust. The winning algorithm, e.g., is predominantly dependent on price-sensitive customers that can easily be targeted by other competitors, whilst other competitors earn their revenue from a more loyal customer base. Fourth, the results reveal that ignoring competition is increasingly harmful when competition is more fierce, i.e., when the number of competitors in the market is large and/or price sensitivity of the customers is high. Finally, the analysis reveals that too much exploration can hurt performance significantly.

Possible extensions to this study that could enhance its generalizability, is to impose

more complex market dynamics in the simulations, such as temporal dependencies or strategic customer behavior. Although it is appealing to do this, we chose not to do so, since it makes it more cumbersome to relate the algorithms and their performance to the dynamics of the market. Another interesting extension is that of a market in which competitors offer assortments of products, and can change both prices and the composition of their assortments over time.

3.A Appendix: Competitor Algorithm Descriptions

Competitor LOGIT

This competitor models the demand according to a finite mixture logit model, where the mixture is taken over the number of possible customer arrivals. Thus, a probability distribution over the number of arrivals in a single period is estimated and for each possible number of arrivals, a different multinomial logit model is estimated as well. Each multinomial logit model here, induces a probability distribution over the competitors, i.e., it specifies with which probability an arriving customer purchases from each competitor (including a no-purchase option). In doing so, it is assumed that the utility of buying from competitor i is of the form $a - bp_i$, where p_i is the price posted by competitor i and a and b are assumed to differ across the mixture components.

In practice, this competitor uses the first 100 time periods to estimate the maximum number of arriving customers in a single time period. This is done by setting a price of 0 for the first period and for each of the following 99 periods of this exploration phase, the price is set as the minimum of the prices observed in the previous period. After these 100 periods, an upper bound on the number of arrivals in a single period is taken as the maximum realized demand in a single period multiplied by $(m+1)$, i.e., the number of competitors plus one. Subsequently, an Expectation-Maximization algorithm is used to estimate a probability distribution over the number of arrivals, as well as the parameters of the multinomial logit models. All these parameters are updated every 20 time periods.

To optimize prices, in every period the competitors' prices for the period to come are predicted. For this purpose, it is assumed that the sorted prices of the competitors follow a multivariate normal distribution, where the sorted prices are used to mitigate the effect of price symmetries. Subsequently, 1000 competitor prices are sampled from the multivariate normal distribution and the revenue function is approximated by averaging over these realizations. To optimize the price, a crude line search over a dis-

cretization of the assumed price space $(0, 100)$ is executed and the price with the highest revenue is chosen.

Competitor OLS

The approach of this competitor to pricing is to favor simplicity. The view is taken that competitors' actions cannot be controlled and that for all intents and purposes, they are random. Thus, they are modeled as an aggregate source of random "noise" and the focus is on how the competitor's own price influences demand in this environment. The algorithm is split into an exploration segment and a "running" segment. The exploration segment lasts for the first 40 periods and the running segment lasts for the rest of the 960 periods.

In the exploration segment, the algorithm explores the field to ensure sufficient variation in data. In each period, a price is sampled uniformly from the interval $(0, 100)$. After the exploration period, the algorithm enters the running segment. In the running segment, the majority of the time consists of estimating a demand curve based only on the competitor's own historical prices and optimizing accordingly. To do so, four linear regression models are fit, taking all combinations of log-transformation of both independent (price) and dependent (demand) variables, and the model with the highest R^2 value is chosen (OLS is an acronym for ordinary least squares). Using this model, the price is optimized using a crude line search and, subsequently, a small perturbation is added to the price for further exploration.

Finally, in each period in the running segment there is a 5% chance of further exploration and a 1% chance of "competitive disruption". Here, "competitive disruption" is an action designed to intentionally confuse competitors who attempt to predict competitor prices or who use competitor prices in their model. When this action is initiated the model sets the price to zero in an attempt to confuse competitors via extreme actions.

Competitor B-GRID

B-GRID is adapted from the ε-greedy multi-armed bandit algorithm (Sutton and Barto, 1998). It assumes a bandit framework with ten arms, where the arms pertain to the prices $10, 20, \ldots, 100$ (B-GRID is an acronym for bandit on a grid). Thus selecting the first arm means posting a price of 10. This algorithm neglects competition and simply keeps track of the average revenue under each arm. With probability ε, an arm is selected randomly, whereas with probability $1 - \varepsilon$, the arm that has the highest observed average

revenue is selected. The exploration parameter ε is set to 0.2, so that on average 200 time periods are used for exploration and 800 for exploitation.

Competitor B-BUCKET

This competitor considers the problem of learning and pricing in a multi-armed bandit framework similar to that of B-GRID. In doing so, the optimal price is assumed to be contained in the interval $(0, 100]$, which is split into ten intervals of even length, i.e., it is split into price buckets $(0, 10], (10, 20], \ldots, (90, 100]$. Each of these price buckets pertains to one arm and selecting a specific arm means posting a price that is uniformly sampled from the corresponding price bucket (B-BUCKET is an acronym for bandit with buckets).

To incorporate the competitors' prices, it is assumed that the arms' values, i.e., revenues, depend on the prices posted by the other competitors. More precisely, in each time period the competitors' modal price bucket is forecast using exponential smoothing. The modal price bucket is the bucket that is predicted to contain most of the competitors' prices. We assume that the optimal choice of price to offer is dependent on this modal price bucket.

In practice this works as follows. At each time step, with probability ε an exploration step is performed in which an arm is selected randomly. Alternatively, with probability $1 - \varepsilon$, an exploitation step is undertaken. In this case, the algorithm selects a price from the price bucket with the highest observed average revenue for the predicted modal price bucket. The exploration parameter ε is set to 0.2, so that on average 200 time periods are used for exploration and 800 for exploitation.

Competitor B-MODEL

This competitor advocates a bandit formulation of the problem as well, although its design differs conceptually from that of B-BUCKET and B-GRID. Where the aforementioned two competitors assign prices (or prices buckets) to arms, here, an arm pertains to a demand model (B-MODEL is an acronym for bandit with models). The demand models that constitute the four arms are the following:

– **Demand Model 1** (Bargain hunters) assumes that the distribution of customers' willingness to pay (WTP) is normally distributed and that customers select a competitor's price from the subset of prices that fall below their WTP with probability proportional to $\left(\frac{WTP - p_i}{WTP}\right)^b$, where p_i is the price being offered by competitor i and

b is a parameter that influences customers' price sensitivity. High (low) values of $b > 1$ (< 1) capture customer populations that are highly (in)sensitive to prices close to their reserve price. In general this first demand model captures bargain hunters as in all cases customers will tend to choose low prices where possible.

– **Demand Model 2** (Quality seekers) is a variant of the first demand model but the reserve price of a customer is proportional to $\left(1 - \frac{WTP - p_i}{WTP}\right)^c$. This model captures customers who use price as an indicator of quality. The parameter c has a similar interpretation to b above.

– **Demand Model 3** (Cheapest price subset) assumes that each customer sees a different random subset of the available prices. Customers are assumed to select the cheapest price that is visible to them. The subset is assumed to include a random number of options uniformly distributed between d and e, which are parameters that can be estimated from the demand data.

The fourth arm alone is used for the first 100 time periods with a relatively high exploration rate to provide sufficient data for estimating the parameters a, b, c, and d of the three demand models by means of simulated annealing. After 100 time periods the reward vectors are reset and the four-armed bandit assumes control of pricing. Similar to the previous two bandit algorithms, with probability ε an arm is selected randomly and otherwise the most profitable arm is selected.

Optimal prices are chosen based on a forecast of the competitor price (duopoly) or the profile of competitor prices (oligopoly), where we define the competitor price profile to be an ordered list of competitors' prices. For the oligopoly the competitor price profile is forecast for the next time period using exponential smoothing with trend. In order to estimate the optimal price to charge under each demand model, the algorithm generates a set of potential prices and the projected revenue is evaluated at each price, for the forecast competitor price profile. The price with the highest predicted revenue is assumed to be the best price for this demand model. In an exploitation step, the algorithm selects the arm with the highest predicted revenue and offers the best price for this arm.

Competitor ML

The approach of this competitor is to rely on machine learning techniques to predict demand and optimize prices accordingly (ML is an acronym for machine learning). Much emphasis is put on learning the demand characteristics, as the algorithm dynamically switches back and forth from exploration to exploitation mode over time. In exploration

mode, during forty time periods, prices are set according to a cosine function around the mean price level observed to test a variety of price levels and, possibly, confuse competitors. After this learning cycle, demand is modeled using own prices and the competitor prices as covariates by means of a variety of regression models (least-squares, ridge regression, Lasso regression, Bayesian ridge regression, stochastic gradient descent regression, and random forest) and the best model, in terms of demand prediction, is selected through cross-validation.

Subsequently, the model of choice is used during an exploitation cycle of variable length: the length is sampled uniformly between 70 and 150, however, if the revenue earned deteriorates too fast, then, immediately a new exploration cycle is initiated. The price is optimized by discretizing the price space and computing the revenue for all prices. When a new exploration cycle starts, so either when the exploitation cycle was finished or because the revenue deteriorated significantly, all historical data is disregarded for the benefit of capturing shifts and shocks in the market most adequately.

Competitor GREEDY

This competitor advocates a particularly simple strategy: set the price as the minimum price observed in the previous time period. To avoid a "race to the bottom" with another competitor, the following facility is implemented: if the minimum price observed in the previous period is lower than the 10% percentile of all the prices observed in the last 30 time periods, then the price for the coming period is set as the maximum of this percentile and 5 (i.e., if this 10% percentile is smaller than 5, the price is set to 5).

Competitor WLS

The characterizing feature of this competitor is that it aims to maximize own revenue relative to its competitors. More precisely, it attempts to maximize own revenue minus the revenue of the competitor that earns the most revenue. In doing so, it is assumed that demand of competitor k equals $d(p_k, p_{\bar{k}})$, where p_k is the price of competitor k, $p_{\bar{k}}$ is the $(m-1)$-vector with the prices of the competitors of k, and where the notion of time is suppressed. In addition, it is assumed that $d(\cdot, \cdot)$ is independent of permutations in its second argument, i.e., in the vector $p_{\bar{k}}$. Thus, this algorithm aims to obtain the price that maximizes own revenue compared to the competitors, that is to solve in each time

step,

$$\max_{p_1}\{p_1 d(p_1, p_{\bar{1}}) - \max\{p_k d(p_k, p_{\bar{k}}) \mid k = 2, \ldots, m\}\}. \tag{3.2}$$

where the competitors are indexed 1 to m (the number of competitors) and WLS is indexed 1. Note that $p_1 \in p_{\bar{k}}$ for $k \in \{2, \ldots, m\}$.

The demand function is assumed to be of the form $d(x, y) = a + bx + c\sum_{k=1}^{m-1} y_k$ and the parameters a, b, and c are estimated using weighted least squares (hence the name WLS). To capture different time-dependent aspects of demand, various schemes for the weighting of observations are considered and evaluated based on the Median Absolute Error of their historical demand predictions. The best weighting scheme is used in (3.2) to optimize the price. For this purpose, the price for competitor k in the period to come is predicted based on the median of the historical prices over some window, where the window length is chosen to minimize the Median Absolute Error of historical price predictions.

Finally, for the purpose of exploration, during the first ten periods prices are randomized to guarantee sufficient variance in the observations to estimate the demand models. In addition, when after these ten periods this competitor's own price is constant for three subsequent periods, the prices are randomized for the next period to induce exploration.

Chapter 4

Bayesian Modeling of Customer Choice Across Product Categories for Promotion Optimization

4.1 Introduction

4.1.1 Background and motivation

Consider an apparel retailer that intends to offer personalized promotions to its customers. This retailer would have to address several issues, such as deciding on which garments to offer to each customer and how much to discount these garments. In addition, one may ask the more profound question of whether it is beneficial to offer personalized promotions at all. Such questions concern the notion of customer heterogeneity and the extent to which customer heterogeneity can be understood and leveraged to the benefit of the retailer. This has become a popular subject in scientific research in the fields of economics, marketing, and management science, especially since customer-level purchasing data has become increasingly available.

In this chapter we model how customers choose which products to buy in different product categories. Here, a product category is understood to be a selection of similar products, such as "men's pullovers" at an apparel retailer or "dairy products" at a grocery store. Our study concerns customer-level purchasing data of a leading Dutch department store. We estimate how brand and price drive the decision-making process on a customer level by relying on the rich literature concerning customer choice modeling. More precisely, for each customer, we assess price sensitivity and brand preferences per category, and the extent to which these are similar across product categories. This is relevant for targeted marketing, which we illustrate by presenting an algorithm that

leverages customer-level transaction data to optimize personalized promotions. This optimization problem consists of selecting the *right* combination of a product and corresponding discount level to offer as a personalized promotion to a customer.

4.1.2 Related literature

Multi-category shopping behavior has been an active field of research since the late 1990s. This can mainly be attributed to the increased availability of consumer grocery shopping data through data providers, such as IRI and A.C. Nielsen, as well as the development of appropriate methodology from the field of Bayesian statistics. One of the main reasons is that it has been recognized that single-category models, which ignore the cross-category dependencies studied in multi-category models, may lead to a biased understanding of customer shopping behavior (Russell and Kamakura, 1998).

One particular stream of research focuses on modeling in which categories a customer makes purchases on a given shopping trip—so-called category purchase incidence (e.g., Manchanda et al., 1999, Chib et al., 2002). These studies are mainly concerned with the dependency between categories when deciding which categories to buy, e.g., because categories are considered complements, such as pasta and pasta sauce. These studies do not take into account the actual product that is purchased within a certain category. We refer to Seetharaman et al. (2005) for an exhaustive overview of multi-category models.

Our work is more related to a stream of research that models brand (or product) choice within categories and the dependencies in preferences across categories. Early work is from Ainslie and Rossi (1998) and Erdem and Winer (1998). The former model the effect of price and promotional efforts on brand choice across five product categories (ketchup, peanut butter, stick margarine, toilet tissue, and canned tuna fish). The results show that there is a positive correlation across categories in price and promotion sensitivity of customers. The latter assess the dependency on brand preference across the categories toothpaste and toothbrushes and find similar results. In a study on breakfast foods and table syrup, Iyengar et al. (2003) investigate models that account for different sources of incomplete information in a multi-category setting. For example, they assess how multi-category models can be used to estimate preferences in a category in which no purchases were observed, by leveraging purchases observed in other categories. In addition, in an attempt to address the curse of (parameter) dimensionality that often frustrates the models discussed here, Singh et al. (2005) introduce methodology in which the preferences for product attributes are projected onto a low-dimensional space. The model is empirically validated on both related categories (potato chips, tor-

tilla chips, and pretzels) and unrelated categories (sliced cheese and mayonnaise). More recently, Duvvuri et al. (2007) and Ma et al. (2012) study the effects of price sensitivity and brand choice, respectively, across categories that are often bought contemporaneously (i.e., *complementary* categories) by using data from, amongst others, cake mix and cake frosting.

4.1.3 Contributions

Our work is distinctive from the existing literature in several ways. First of all, the previous overview of existing literature reveals that—at least to our best knowledge—modeling dependencies in product choice behavior across categories has exclusively been considered in the context of grocery retailing. Presumably, this is due to the availability of high-quality transaction data in the grocery retailing industry. In the current work, however, we consider purchasing behavior of durable goods (namely, apparel products). A notable difference with respect to grocery retailing regards the fact that in apparel retailing customers frequently leave the store without buying anything (which remains unobserved), as opposed to grocery retailing. Consequently, we model customer choice given that a customer purchases in a category, whereas in grocery retailing one is able to condition on a visit to the store. We show empirically that in our setting it is still possible to infer dependencies in purchasing behavior across categories. The results show that a customer's price sensitivity and brand preferences carry over across categories, as has previously been observed in grocery retailing.

Second, for estimation and prediction, our model does not require that each customer has purchased in all categories (we refer to this as the data being *sparse*). This means that we map a customer's revealed preferences (i.e., purchases) to parameter estimates for categories in which no purchases were observed. Therefore, our model can predict behavior in categories in which customers did not purchase before. This makes our work complementary to that of Iyengar et al. (2003) on multi-category buying in grocery retailing (see also Section 4.1.2).

Another distinctive feature of our work relates to the construction of choice sets, which are defined as the set of alternatives available to a customer when making a purchase. These choice sets are required in customer choice modeling to estimate a customer's preferences for certain product characteristics (such as price or brand). Given that we only observe purchases, it is not trivial to infer (for each purchase incidence) the choice set. To address this, we construct a procedure that approximates the continuously changing product assortment of the retailer. This is much less of an issue for grocery

retailers, who generally have a much more stable product assortment.

Finally, we show in a stylized setting how our framework can be used to determine, for a given category, which combination of product and discount factor maximizes the expected revenue. The results indicate that discounts should be given to customers that are more price sensitive. In addition, discounts should be mostly given on products that are not highly valued by the respective customers. As such, price discounts are used as a tool to persuade customers to purchase products that likely would not have been purchased otherwise.

4.1.4 Outline

This chapter is organized as follows. First, in Section 4.2, we present the details of the hierarchical Bayesian model that we use in this study. Then, in Section 4.3, we present a framework to sample from the joint distribution of model parameters (conditional on the data). In Section 4.4, we present the results of our empirical study and Section 4.6 describes an application to promotion optimization. In Section 4.5, we conduct a simulation study to assess the performance of our sampling framework. Finally, in Section 4.7 we conclude and provide directions for future research.

4.2 Model Specification

We assume that a customer, when choosing a product, latently assigns utility to the alternatives available and, subsequently, purchases the product that maximizes the customer's utility. This assumption is fundamental to the field of customer choice modeling and, more generally, to the paradigm of *random utility* in economics (see, e.g., Train, 2009, Chapter 2). Specifically, we model customer choice behavior conditional on a category purchase, i.e., given that a customer purchases in a given category, we model the likelihood with which a certain alternative within that category is selected based on the composition of the assortment and the characteristics of the alternatives available.

To formalize the notion of utility maximization, let $n \in \mathbb{N}$ be the number of customers available in the dataset and let $m \in \mathbb{N}$ be the number of product categories. For each $i \in \{1, \dots, n\}$ and $c \in \{1, \dots, m\}$, let $\mathcal{P}_{ic}$ be the set of products available to customer i in category c. Furthermore, let $k \in \mathbb{N}$ be the number of product attributes and, for each $i \in \{1, \dots, n\}$, $c \in \{1, \dots m\}$, and $j \in \mathcal{P}_{ic}$, let $x_{ijc} \in \mathbb{R}^k$ be the vector of product attributes that defines product j in category c for customer i (e.g., price, brand, and color).

Then, for each $i \in \{1, \ldots, n\}$, $c \in \{1, \ldots, m\}$, and $j \in \mathcal{P}_{ic}$, we assume that the random utility that customer i assigns to product j from category c is equal to

$$U_{ijc} := \beta_{ic}^{\top} x_{ijc} + \varepsilon, \tag{4.1}$$

where ε is i.i.d. standardized Gumbel distributed (i.e., $\mathbb{P}(\varepsilon \leq t) = \exp(-\exp(-t))$ for all $t \in \mathbb{R}$) and β_{ic} is a random vector with attribute weights of length k.

To allow for customer heterogeneity and to capture correlation in preferences for attributes across categories, we assume that the attribute weights follow a multivariate normal distribution:

$$\beta_i := \left(\beta_{i1}^{\top}, \beta_{i2}^{\top}, \ldots, \beta_{im}^{\top}\right)^{\top} \sim N(\mu, \Sigma), \text{ for each } i \in \{1, \ldots, n\}, \tag{4.2}$$

where μ and Σ are of dimension km and $km \times km$, respectively, and pertain to the expectation and covariance of the multivariate normal distribution (recall that k is the number of product attributes that defines a product and m is the number of categories).

The specification in (4.2) is flexible, since it captures dependencies between all attributes of all products in all categories. For example, Σ measures the extent to which price sensitivities in one category carry over to other categories. Similarly, we expect that if a customer enjoys relatively high utility from a specific brand in a certain category, this customer also favors this brand in other categories. In Section 4.6, we show how such information can be leveraged for the purpose of offering personalized promotions. In addition to flexibility, the normality assumption on β_i makes the model tractable, since it allows for efficient inference (we further elaborate on this in Section 4.3).

Our distributional assumption in (4.2) differs from, e.g., Ainslie and Rossi (1998) and Singh et al. (2005), which use a decomposition of Σ to control for its dimensionality. In this work, we allow for an unrestricted covariance matrix Σ, since its estimation does not result in any problems when applied to our dataset and the results prove to be satisfying.

For each $i \in \{1, \ldots, n\}$ and $c \in \{1, \ldots, m\}$, it follows from (4.1) that—given that customer i makes a purchase in category c—the conditional probability that customer i chooses product $j \in \mathcal{P}_{ic}$ is equal to (Train, 2009):

$$\mathbb{P}\left(\max_{j' \in \mathcal{P}_{ic}} U_{ij'c} = U_{ijc} \,\middle|\, \beta_{ic}\right) = \frac{\exp\left(\beta_{ic}^{\top} x_{ijc}\right)}{\sum_{j' \in \mathcal{P}_{ic}} \exp\left(\beta_{ic}^{\top} x_{ij'c}\right)}, \tag{4.3}$$

which is well-defined, since ties occur with probability zero.

The model specification in (4.1)-(4.2) with corresponding choice probabilities in (4.3) is known as a mixed multinomial logit (MML) model or random coefficients logit model. In case the random parameters (β_{ic}) are assumed to be deterministic, we obtain the multinomial logit (MNL) model, which is arguably the most famous customer choice model and has been applied in many research areas and in practice (see, e.g., Train, 2009).

4.3 Model Estimation

4.3.1 Prior distributions

We adopt a Bayesian approach for the purpose of doing statistical inference on the model introduced in the previous section. To this end, we consider the parameters of the multivariate normal distribution in (4.2), namely μ and Σ, to be random variables themselves. Due to the hierarchical structure of parameters, this approach is referred to as hierarchical Bayesian modeling.

The objective then is to obtain distributions (or rather samples from distributions) of $\{\beta_i\}_{i=1,\dots,n}$, μ, and Σ conditional on the data that we observe (these conditional distributions are called *posterior* distributions in Bayesian terminology). The posterior distributions of $\{\beta_i\}_{i=1,\dots,n}$ are useful for customer-level inference, as well as for targeted marketing, as in our application of personalized promotions.

Taking a standard approach in Bayesian statistics (e.g., Rossi et al., 2012), we impose the following prior distributions on μ and Σ:

$$\Sigma \sim \text{IW}(\nu, V) \text{ and } (\mu \mid \Sigma) \sim N(\overline{\mu}, \sigma \cdot \Sigma), \tag{4.4}$$

where IW denotes the Inverse Wishart distribution and where $V \in \mathbb{R}^{km \times km}$, $\nu \in (km, \infty)$, $\overline{\mu} \in \mathbb{R}^{km}$, and $\sigma \in \mathbb{R}_{++}$ are hyperparameters.

The prior distributions in (4.4) are so-called conditionally conjugate priors, which allows for efficient inference through Gibbs sampling, as will be shown below in Section 4.3.3. Before doing so, in the following section we first formalize the information (i.e., transactions) that we observe.

4.3.2 Observables

For each $i \in \{1, \ldots, n\}$, let $\tau_i \in \mathbb{N}$ be the total number of products purchased by customer i (i.e., these are historically observed). In addition, with slight abuse of notation, for each $i \in \{1, \ldots, n\}$, $c \in \{1, \ldots, m\}$, and $t \in \{1, \ldots, \tau_i\}$, we denote by $\mathcal{P}_{ict}$ the set of products available to customer i in category c at the time of its t^{th} purchase and, for each $j \in \mathcal{P}_{ict}$, we let

$$y_{ijct} = 1$$

if customer i purchased product j from category c at purchase incidence t and 0 otherwise. Observe that this implies that $\sum_{c=1}^{m} \sum_{j \in \mathcal{P}_{ict}} y_{ijct} = 1$ for each $i \in \{1, \ldots, n\}$ and $t \in \{1, \ldots, \tau_i\}$. In addition, we do not require that, for each $i \in \{1, \ldots, n\}$ and $c \in \{1, \ldots, m\}$, it holds that

$$\sum_{t=1}^{\tau_i} \sum_{j \in \mathcal{P}_{ict}} y_{ijct} > 0.$$

That is, we do not require that each customer made at least one purchase in each category.

Abusing notation once more, for each $i \in \{1, \ldots, n\}$, $c \in \{1, \ldots, m\}$, $t \in \{1, \ldots, \tau_i\}$, and $j \in \mathcal{P}_{ict}$, we denote by $x_{ijct} \in \mathbb{R}^k$ the attributes of product j in category c when customer i made its t^{th} purchase. Finally, for each $i \in \{1, \ldots, n\}$, we denote *all the data* pertaining to customer i by

$$z_i := \left\{ (y_{ijct},\, x_{ijct}) \;\middle|\; j \in \mathcal{P}_{ict},\, t \in \{1, \ldots, \tau_i\},\, c \in \{1, \ldots, m\} \right\}.$$

4.3.3 Gibbs sampling

For the purpose of doing Bayesian inference, our objective is to sample from the joint probability distribution function (pdf)

$$f\left(\beta_1, \ldots, \beta_n, \mu, \Sigma \,|\, z_1, \ldots, z_n, \theta\right), \tag{4.5}$$

where $\theta := (\nu, V, \overline{\mu}, \sigma)$ are the hyperparameters and, as is common use in Bayesian statistics, the arguments of $f(.|.)$ are used to distinguish between pdf's. These samples are then used to compute statistics (e.g., the mean of parameters) and to analyze marginal distributions.

Given that the joint distribution in (4.5) is high-dimensional (at least, when the number of customers is large) and of non-standard form, it is computationally intractable to

sample from it directly a sufficient number times. To alleviate this, we rely on Markov chain Monte Carlo (MCMC) methods, which work by constructing a Markov chain that has the distribution of interest (the *target* distribution in Bayesian terminology) as its stationary distribution (see, e.g., Gelman et al., 2013). The general idea is to sequentially sample in regions of the distribution with relatively high probability mass, which make these methods capable of sampling efficiently from high-dimensional distributions.

In particular, we rely on the Gibbs sampling algorithm, which sequentially samples from subsets of the parameters. In our case, this works as follows. Let $\zeta \in \mathbb{N}$ be the number of required samples from (4.5). Then, for $s = 1, \ldots, \zeta$, we sequentially sample

$$\beta_i^{(s)} \,\Big|\, z_i, \mu^{(s-1)}, \Sigma^{(s-1)} \qquad \forall i \in \{1, \ldots, n\} \tag{4.6}$$

$$\Sigma^{(s)} \,\Big|\, \beta_1^{(s)}, \ldots, \beta_n^{(s)}, \mu^{(s-1)}, \theta \tag{4.7}$$

$$\mu^{(s)} \,\Big|\, \beta_1^{(s)}, \ldots, \beta_n^{(s)}, \Sigma^{(s)}, \theta, \tag{4.8}$$

where the superscripts are used to index samples of the corresponding variables and $\mu^{(0)}$ and $\Sigma^{(0)}$ are, e.g., initialized randomly.

It can be shown that the sequence

$$\left(\beta_1^{(1)}, \ldots, \beta_n^{(1)}, \mu^{(1)}, \Sigma^{(1)}\right), \ldots, \left(\beta_1^{(\zeta)}, \ldots, \beta_n^{(\zeta)}, \mu^{(\zeta)}, \Sigma^{(\zeta)}\right)$$

constructed through (4.6)-(4.8) is a Markov chain that has a stationary distribution equal to the joint distribution from (4.5) (Gelman et al., 2013). In practice, samples generated by the Gibbs sampler are initially discarded—the so-called *burn-in* period—to let the Markov chain reach its stationary distribution. Only samples generated after this burn-in period are then used.

The last two Gibbs *layers*, i.e., (4.7) and (4.8), consist of well-documented steps that rely on standard Bayesian theory on MCMC for multivariate regression models and are described in the appendix. In (4.6), however, we require a method to sample β_i for all $i \in \{1, \ldots, n\}$ from its conditional posterior distribution. Given that this distribution has a non-standard density that is only known up to a constant, we use another MCMC algorithm, namely Metropolis-Hastings (MH), to generate samples (we elaborate on this in the following section). In doing so, we use an MCMC algorithm (namely, MH), within another MCMC algorithm (namely, Gibbs), which is often referred to as a Metropolis-Hastings-within-Gibbs algorithm. We provide our proposed sampling scheme for step (4.6) in the subsequent section.

4.3.4 Metropolis-Hastings with sparse data

Let $i \in \{1, \ldots, n\}$, $c \in \{1, \ldots, m\}$, and $s \in \{2, \ldots, \zeta\}$. For ease of exposition, but without loss of generality, suppose that customer i has only made purchases in category c (recall that we refer to this as the data being sparse). The objective is to obtain a sample $\beta_i^{(s)}$ from the distribution of β_i conditional on z_i, $\mu^{(s-1)}$, and $\Sigma^{(s-1)}$, cf. (4.6). To this end, we split step (4.6) in the following two sub-layers:

$$\beta_{ic}^{(s)} \,\Big|\, z_i, \mu^{(s-1)}, \Sigma^{(s-1)} \tag{4.9}$$

$$\beta_{i1}^{(s)}, \ldots, \beta_{i(c-1)}^{(s)}, \beta_{i(c+1)}^{(s)}, \ldots, \beta_{im}^{(s)} \,\Big|\, z_i, \mu^{(s-1)}, \Sigma^{(s-1)}, \beta_{ic}^{(s)}. \tag{4.10}$$

For the sampling step in (4.9), we rely on an MH algorithm, which is an MCMC algorithm that allows to sample from distributions that are only known up to a constant. This suits our purpose, since the distribution corresponding to (4.9) is only known up to a constant:

$$\begin{aligned} f\left(\beta_{ic} \,\middle|\, z_i, \mu^{(s-1)}, \Sigma^{(s-1)}\right) &\propto f\left(z_i \mid \beta_{ic}\right) f\left(\beta_{ic} \,\middle|\, \mu^{(s-1)}, \Sigma^{(s-1)}\right) \\ &= \prod_{t=1}^{\tau_i} \prod_{j \in \mathcal{P}_{ict}} \left(\frac{e^{\beta_{ic}^\top x_{ijct}}}{\sum_{j' \in \mathcal{P}_{ict}} e^{\beta_{ic}^\top x_{ij'ct}}} \right)^{y_{ijct}} \cdot \phi\left(\beta_{ic} \,\middle|\, \mu_c^{(s-1)}, \Sigma_c^{(s-1)}\right), \end{aligned} \tag{4.11}$$

where $\phi\left(\cdot \mid \mu, \Sigma\right)$ denotes the pdf of the multivariate normal distribution with mean μ and covariance Σ and $\mu_c^{(s-1)}$ and $\Sigma_c^{(s-1)}$ denote the k elements of $\mu^{(s-1)}$ and the $k \times k$ sub-matrix of $\Sigma^{(s-1)}$ that pertain to category c.

The MH algorithm works sequentially by generating a candidate sample for $\beta_{ic}^{(s)}$ based on the current sample $\beta_{ic}^{(s-1)}$ (thus forming a Markov chain) and, with a certain probability, accepting this candidate and otherwise rejecting it. In case it is accepted, the candidate sample becomes $\beta_{ic}^{(s)}$ (the next sample in the chain) and in case of rejection the next sample is a repetition of $\beta_{ic}^{(s-1)}$ (the current sample). The acceptance probability is (partly) determined by the ratio of the likelihood in (4.11) evaluated at the candidate sample to the likelihood in (4.11) evaluated at the current sample (so that candidate samples from regions with high probability mass are more likely to be accepted).

The candidate samples are generated according to a so-called proposal distribution, which can be set at our own discretion (e.g., adding random Gaussian noise to the current sample is a popular choice, leading to the so-called random walk sampler). In case the candidate samples are too close to the previous samples, the Markov chain explores the target distribution very slowly, leading to an inefficient sampling algorithm (a *sticky*

chain in the MCMC lingo). On the other hand, when candidates are sampled relatively far from the current sample, the Markov chain tends to *overshoot* regions of high probability mass, leading to many candidates being rejected and, consequently, to inefficient sampling as well.

To sample from (4.11), we adopt a MH algorithm that generates candidate samples based on the current sample by adding a normally distributed increment with a mean proportional to the gradient of the pdf at the current position of the chain. This implies that every jump of the chain is biased in the direction of the gradient of the target distribution, thereby sampling towards areas of high probability mass. The algorithm is called the truncated Metropolis-adjusted Langevin algorithm (TMALA) (Roberts et al., 1996, Atchade, 2006) and works as follows.[1]

Let $\kappa \in \mathbb{R}_{++}$ be a hyperparameter and, for all $\tilde{\beta}_{ic} \in \mathbb{R}^k$, let

$$\delta\left(\tilde{\beta}_{ic}\right) := \frac{\kappa}{\kappa \vee \left\|\nabla \log f\left(\tilde{\beta}_{ic} \,\middle|\, z_i, \mu^{(s-1)}, \Sigma^{(s-1)}\right)\right\|_2} \nabla \log f\left(\tilde{\beta}_{ic} \,\middle|\, z_i, \mu^{(s-1)}, \Sigma^{(s-1)}\right), \tag{4.12}$$

where $\nabla \log f(\tilde{\beta}_{ic}|\cdot)$ denotes the gradient of the log of the posterior distribution of β_{ic} from (4.11) evaluated at $\tilde{\beta}_{ic}$ and where we observe that $\|\delta(\tilde{\beta}_{ic})\|_2 \leq \kappa$ for all $\tilde{\beta}_{ic} \in \mathbb{R}^k$. Then, we sample $\beta_{ic}^{(s)}$ from (4.11) by utilizing the following scheme, where $\gamma_i \in \mathbb{R}_{++}$ is a hyperparameter.

1. Sample $\tilde{\beta}_{ic}$ randomly according to the following proposal distribution:

$$N\left(\beta_{ic}^{(s-1)} + \frac{\gamma_i}{2}\Sigma^{(s-1)}\delta\left(\beta_{ic}^{(s-1)}\right), \gamma_i \Sigma^{(s-1)}\right). \tag{4.13}$$

2. Set

$$\alpha \leftarrow 1 \wedge \frac{f\left(\tilde{\beta}_{ic} \,\middle|\, z_i, \mu^{(s-1)}, \Sigma^{(s-1)}\right)}{f\left(\beta_{ic}^{(s-1)} \,\middle|\, z_i, \mu^{(s-1)}, \Sigma^{(s-1)}\right)} \frac{\phi\left(\beta_{ic}^{(s-1)} \,\middle|\, \tilde{\beta}_{ic} + \frac{\gamma_i}{2}\Sigma^{(s-1)}\delta\left(\tilde{\beta}_{ic}\right), \gamma_i\Sigma^{(s-1)}\right)}{\phi\left(\tilde{\beta}_{ic} \,\middle|\, \beta_{ic}^{(s-1)} + \frac{\gamma_i}{2}\Sigma^{(s-1)}\delta\left(\beta_{ic}^{(s-1)}\right), \gamma_i\Sigma^{(s-1)}\right)}. \tag{4.14}$$

3. With probability α, set

$$\beta_{ic}^{(s)} \leftarrow \tilde{\beta}_{ic},$$

[1] Initially, we implemented the standard random walk MH algorithm, however, this led to very slow convergence to the stationarity distribution, where, as a convergence condition, we used the Gelman-Rubin diagnostic (Gelman and Rubin, 1992) and assumed that the chain converged to the stationary distribution if the Gelman-Rubin diagnostic was at most 1.35 for all elements of μ and Σ.

with probability $1 - \alpha$, set

$$\beta_{ic}^{(s)} \leftarrow \beta_{ic}^{(s-1)}.$$

In (4.14), the acceptance probability α is constructed such that the resulting Markov chain has a stationary distribution and that this distribution is equal to the target distribution in (4.11).

For each $i \in \{1, \ldots, n\}$, the parameter $\gamma_i \in \mathbb{R}_{++}$ scales the mean and covariance of the proposal function in (4.13). For $\gamma_i \searrow 0$, the proposal distribution is degenerate with all probability mass at $\beta_{ic}^{(s-1)}$. For larger values of γ_i, the variance of the proposal distribution increases and the mean shifts towards the gradient of the target distribution.

After sampling $\beta_{ic}^{(s)}$ according to the previous scheme, we continue to step (4.10). For notational convenience, suppose $c = 1$, i.e., customer i only made purchases in the first category, and partition

$$\mu^{(s-1)} = \begin{pmatrix} \mu_1^{(s-1)} \\ \mu_2^{(s-1)} \end{pmatrix} \text{ and } \Sigma^{(s-1)} = \begin{pmatrix} \Sigma_{11}^{(s-1)}, \Sigma_{12}^{(s-1)} \\ \Sigma_{21}^{(s-1)}, \Sigma_{22}^{(s-1)} \end{pmatrix},$$

where $\mu_1^{(s-1)} \in \mathbb{R}^k$, $\mu_2^{(s-1)} \in \mathbb{R}^{k(m-1)}$, $\Sigma_{11}^{(s-1)} \in \mathbb{R}^{k \times k}$, $\Sigma_{12}^{(s-1)} \in \mathbb{R}^{k \times k(m-1)}$, $\Sigma_{21}^{(s-1)} \in \mathbb{R}^{k(m-1) \times k}$, and $\Sigma_{22}^{(s-1)} \in \mathbb{R}^{k(m-1) \times k(m-1)}$. It follows that step (4.10) is straightforward, since

$$\left(\beta_{i2}, \ldots, \beta_{im} \;\middle|\; z_i, \mu^{(s-1)}, \Sigma^{(s-1)}, \beta_{i1}^{(s)} \right) \tag{4.15}$$

is normally distributed with mean equal to

$$\mu_2^{(s-1)} + \Sigma_{21}^{(s-1)} \left(\Sigma_{11}^{(s-1)} \right)^{-1} \left(\beta_{i1}^{(s)} - \mu_1^{(s-1)} \right)$$

and covariance equal to

$$\Sigma_{22}^{(s-1)} - \Sigma_{21}^{(s-1)} \left(\Sigma_{11}^{(s-1)} \right)^{-1} \Sigma_{12}^{(s-1)}.$$

Through (4.15), we map a customer's revealed preferences to parameters pertaining to categories in which no purchases were made (this is similar to the 'full-demo' scenario in Iyengar et al. (2003)).

4.3.5 Hyper parameter setting

The hyperparameters pertaining to the prior distributions in (4.4) are set to $\nu = 15$, $V = \nu \cdot I_{km}$, $\bar{\mu} = 0_{km}$, and $\sigma = 100$ (where I_{km} and 0_{km} denote the identity matrix and zero vector, respectively, of dimension km). In this way, we induce diffuse (i.e., high variance) prior distributions on the parameters (Rossi et al., 2012) to let the data *speak for itself*. The parameters that pertain to the TMALA sampling scheme in Section 4.3.4, namely γ_i for all $i \in \{1, \ldots, n\}$ and κ, are determined as follows.

During the burn-in phase (i.e., during the first number of iterations used to let the Markov chain reach its stationary distribution), we adaptively tune γ_i for all $i \in \{1, \ldots, n\}$. The objective of doing this is to obtain favorable acceptance rates of the TMALA sampler, i.e., to ensure that neither too many nor too few samples are accepted (see also the discussion on this matter in Section 4.3.4). Theoretically, little is known about what a *good* (let alone optimal) acceptance rate is. One of the few theoretical results on this matter is from Roberts et al. (1997), who prove for a special case that the asymptotically optimal acceptance rate is equal to 0.234. For more exotic cases (such as our Metropolis-Hastings-within-Gibbs scheme), acceptance rates between 0.1 and 0.6 have numerically proven to yield efficient samplers (Rosenthal et al., 2011). Numerical experiments indicate that our sampling scheme is not sensitive to the acceptance rate and that acceptance rates in the range from 0.2 to 0.6 yields satisfying results. Therefore, we construct the following ad-hoc hyperparameter tuning approach:

– During the burn-in phase, each 100^{th} iteration, update γ_i for each $i \in \{1, \ldots, n\}$ as follows:

$$\gamma_i \leftarrow \begin{cases} 1.25 \cdot \gamma_i, & \text{if the acceptance rate over the last 100 draws was } > 0.55, \\ 0.75 \cdot \gamma_i, & \text{if the acceptance rate over the last 100 draws was } < 0.45, \\ \gamma_i, & \text{else.} \end{cases}$$

By adapting γ_i only during the burn-in phase, and not during the actual sampling, we ensure that the Markov chain remains ergodic. Alternative approaches, in which the parameters are adapted during the actual sampling, while maintaining ergodicity, can be found in Rosenthal et al. (2011).

The parameter $\kappa \in \mathbb{R}_{++}$ truncates the length of the log of the gradient of the target distribution to κ, cf. (4.12). In doing so, we prevent the proposal function from overshooting areas of high probability mass. Numerical experiments that we ran confirm that, when we do not truncate, the sampling scheme becomes inefficient exactly be-

cause of this reason. A relatively broad range of values for κ yielded favorable results and we decided to set it to 5.0.

4.4 Empirical Study

4.4.1 Data

We apply our model to transaction data of a Dutch department store. The data spans the years 2011-2013 and regards purchases made by loyalty cardholders, which allows us to attribute each purchase to an individual cardholder. The dataset comprises the four largest brands (measured in terms of purchase incidence) for each of the product categories men's pullovers, men's shirts, and men's polos. We selected 12,000 customers randomly that made at least two purchases in total. In doing so, we avoid biases that occur when only customers are selected whose purchase frequency exceeds a certain high threshold (Kim and Rossi, 1994). For each customer, one purchase was removed from the training data to act as a hold-out sample.

We include brand and price as attributes, which means that we assume that each product is defined by these two attributes. Consequently, two products of the same category, brand, and price are treated as the same product, even if they differ in, e.g., color, fabric or size. An opportunistic approach is to classify each distinguishable product as a separate product in the model. However, this requires the inclusion of all the attributes that uniquely define a certain product. In the current setting, this is intractable given that the number of parameters increases dramatically, while the number of observations per product decreases in the number of product attributes.

The number of customers in the dataset is large relative to the number of observations per customer. This can be observed from Table 4.1, which represents the distribution of purchases across categories and Table 4.2, which presents the number of different categories in which each customer makes a purchase (the brand names are anonymized to guarantee confidentiality). On average, we observe 3.3 purchases per customer.

Although we consider the data cleaning as a vital part of this work, a detailed description is omitted, since it consists of rather straightforward and tedious steps, such as removing duplicates and accounting for returned items and erroneous scanner information.

Table 4.1: The number of purchases observed per category, the number of customers that purchased in the respective categories, and the brands per category.

	Purchases	Customers	Brands
Men's shirts	22,275	8,330	A, B, D, E
Men's pullovers	12,664	6,312	A, B, C, F
Men's polos	4,975	2,900	A, B, C, G

Table 4.2: The number of customers that made a purchase in 1, 2 and 3 different categories, respectively.

Number of different categories bought	1	2	3	
Customers	7,408	3,642	950	12,000

4.4.2 Choice set construction

The construction of the choice sets (i.e., the set of alternatives available each time a purchase was made) is challenging, since we only observe transactions and do not observe which products were available for which price at each moment in time. To this end, we assume a product is available in all sizes from the first day it is purchased, until the day a purchase is observed for the last time. Furthermore, at each point in time, the price is set as the price that was paid when the last purchase was observed.

By constructing the products' availabilities and price paths in this manner, the cardinality of the choice set of each category varies over time (with a mean across all time periods and all categories of 62.0). In doing so, the choice sets are dynamic, which fits the dynamic character of the stock of an apparel retailer. This differs from the multi-category choice studies in grocery retailing mentioned in the literature review in Section 4.1.2, where the choice sets are assumed to be static over time (which, in turn, fits the static character of the stock of a grocery retailer).

We measure prices in multiples of hundreds of euros (e.g., 1.5 pertains to EUR 150) to prevent numerical problems when computing likelihoods, since overflows may occur when computing the sum of the exponential terms in (4.3).

4.4.3 Results

We implemented our sampling algorithms in Python 2.7.9 using NumPy 1.9.2 and, to accelerate computations, we used the Lisa computer cluster (SURFsara, 2015). In total,

Table 4.3: Posterior mean of μ and correlation matrix based on posterior mean of Σ. The capital subscripts pertain to the various brands and p pertains to the price attribute. The standard deviations are provided in the parenthesis.

	Men's pullovers				Men's shirts				Men's polos			
	β_A	β_B	β_C	β_p	β_A	β_B	β_D	β_p	β_A	β_B	β_C	β_p
β_A												
β_B	.34											
	(.06)											
β_C	.58	.60										
	(.04)	(.05)										
β_p	.36	.06	−.17									
	(.05)	(.08)	(.07)									
β_A	.28	−.03	.11	.04								
	(.04)	(.06)	(.04)	(.07)								
β_B	−.13	.22	.06	−.11	.55							
	(.04)	(.06)	(.04)	(.06)	(.02)							
β_D	.38	.04	.23	−.02	.79	.50						
	(.04)	(.07)	(.05)	(.06)	(.01)	(.03)						
β_p	−.19	.07	−.24	.35	.09	.13	−.20					
	(.07)	(.08)	(.07)	(.07)	(.05)	(.05)	(.05)					
β_A	.42	.00	.10	.07	.28	−.01	.31	−.21				
	(.06)	(.09)	(.07)	(.06)	(.04)	(.04)	(.04)	(.08)				
β_B	−.07	.23	−.04	.09	−.11	.20	−.08	.10	.37			
	(.07)	(.09)	(.07)	(.06)	(.05)	(.05)	(.05)	(.08)	(.07)			
β_C	.17	.24	.41	−.17	−.00	.04	.03	−.15	.50	.59		
	(.07)	(.09)	(.08)	(.07)	(.06)	(.05)	(.05)	(.08)	(.06)	(.05)		
β_p	−.14	.39	−.05	.34	−.20	.12	−.28	.51	−.47	.28	−.13	
	(.07)	(.09)	(.08)	(.07)	(.06)	(.06)	(.06)	(.06)	(.04)	(.05)	(.05)	
μ	3.83	1.68	2.18	−1.65	1.39	0.61	0.78	−0.76	−1.94	0.85	0.61	−5.90
	(.18)	(.18)	(.19)	(.04)	(.07)	(.08)	(.08)	(.03)	(.27)	(.17)	(.18)	(.15)

we ran our Markov chain Monte Carlo (MCMC) sampling algorithm depicted in (4.6)-(4.8) for 40,000 iterations, after which we discarded the first 20,000 as burn-in and kept the rest for analysis. Given that the dataset comprises three categories and four brands per category (see Section 4.4.1), it follows that k is equal to four: one attribute pertains to the price and three attributes are dummy variables that indicate a category; one category is set to zero to ensure identifiability of the model. In each iteration, we obtain a sample for μ, Σ, and for β_i for all $i \in \{1, \ldots, n\}$, which corresponds to 144,090 values in our setting. The algorithm is initialized with $\mu^{(0)}$ set to the zero vector and $\Sigma^{(0)}$ set to the identity matrix.

Population parameters μ and Σ

In Table 4.3, the posterior mean of μ and a correlation matrix based on the posterior mean of Σ are presented. Regarding Σ, a number of interesting remarks can be made. First of all, we observe positive correlations between the price attribute parameters across categories. This indicates that price (in)sensitivite of a customer is a trait that exhibits similarly across categories, as one would expect. Except for the fact that this is a satisfying result from an empirical point of view, it is also important for the functioning of our model, since these covariances are used to project a customer's price sensitivity in a certain category onto price sensitivities in categories for which no observations are available. For example, suppose that we observe that a customer is relatively price sensitive in the category men's pullovers and that we observe no purchases in other product categories. Then, according to the results, the model predicts that the customer is also relatively price sensitive in the other two categories (because of the positive correlations of 0.35 and 0.34, respectively). A similar observation can be made regarding preferences for brands. Namely, in all occasions, the correlation between brand-specific constants of the same brand are positive across categories. In addition, in almost all cases the correlation between the same brand in different categories is higher than the correlation with the other available brands.

Regarding μ, we observe that all the price attributes have the expected negative sign, which implies that, on average, the demand for a product decreases in its price. For some individual customers, however, we observe a positive mean price sensitivity parameter. This is presumably due to the fact that in the data, some customers consistently buy at the high end of the price range. Our normality assumption on β_i allows us to quantify the proportion of the population that has a positive price sensitivity parameter. In the category pullovers, for example, the mean of the price parameter is -1.65 (see Table 4.3) and the corresponding variance is 0.54 (this value comes from the non-standardized mean of Σ, which is omitted to preserve space), from which it follows that approximately 1% of the population has price sensitivity parameter with a positive expectation. Furthermore, from the brand-specific constants, one may, for example, infer that, on average, a shirt of brand A is valued roughly EUR 100.- higher than one from brand B.[2]

[2]This holds since $1.39 - 0.76 \cdot 1.00 \approx 0.61$ (recall that the prices are measured in multiples of 100 for numerical stability).

Customer-level parameters

In addition to the population parameters μ and Σ, the samples of $\{\beta_i\}_{i=1,\ldots,n}$ are interesting, as they provide insight into customer-specific preferences and choice behavior. Therefore, to provide an illustration of the kind of results we obtain, we anecdotally discuss one typical customer.[3] For this particular customer, we have observed two purchase incidences, namely two times a pullover of brand A. Both pullovers were bought relatively inexpensively (they were the 13th and 2nd cheapest pullover available at the time of purchase out of 53 and 35 pullovers, respectively). This indicates that this customer is relatively price sensitive. The posterior distributions and means of the parameters of this particular customer are presented in Figure 4.1. From this figure, we observe that the location of the distribution of the parameter pertaining to brand A in the category men's pullovers is relatively high (namely, 4.41, according to the mean), as expected given the purchases made. In addition, we like to emphasize that the distributions of the parameters pertaining to brand A in the categories shirts and polos (in which no observations were available) also have much of their probability mass at higher values (relative to the population average, resembled in μ).

A similar pattern can be observed for the price sensitivity of this customer: since the customer purchases relatively cheap alternatives, we observe relatively high mean price sensitivity in the observed categories pullovers (-2.53 vs -1.65), as well as for the unobserved categories shirts (-1.81 vs -0.76) and polos (-7.07 vs -5.90).

Summarizing, the results are satisfying in a sense that they confirm that customers exhibit cross-category consistency in brand preference and price sensitivity. In addition, we observe that customers are (on average) price sensitive. This is all in line with related work, such as Singh et al. (2005). Furthermore, we have illustrated how our model infers preferences in product categories in which no observations are available.

Clearly, the dependency between attributes across two categories can only be inferred from customers who made purchases in both these categories. It is therefore expected that the sparseness of the data plays an important role in the estimation of the parameters. In the following section, we conduct a simulation study to assess the performance of our framework for various degrees of data sparsity.

[3] This is the customer with ID 2139 in our dataset.

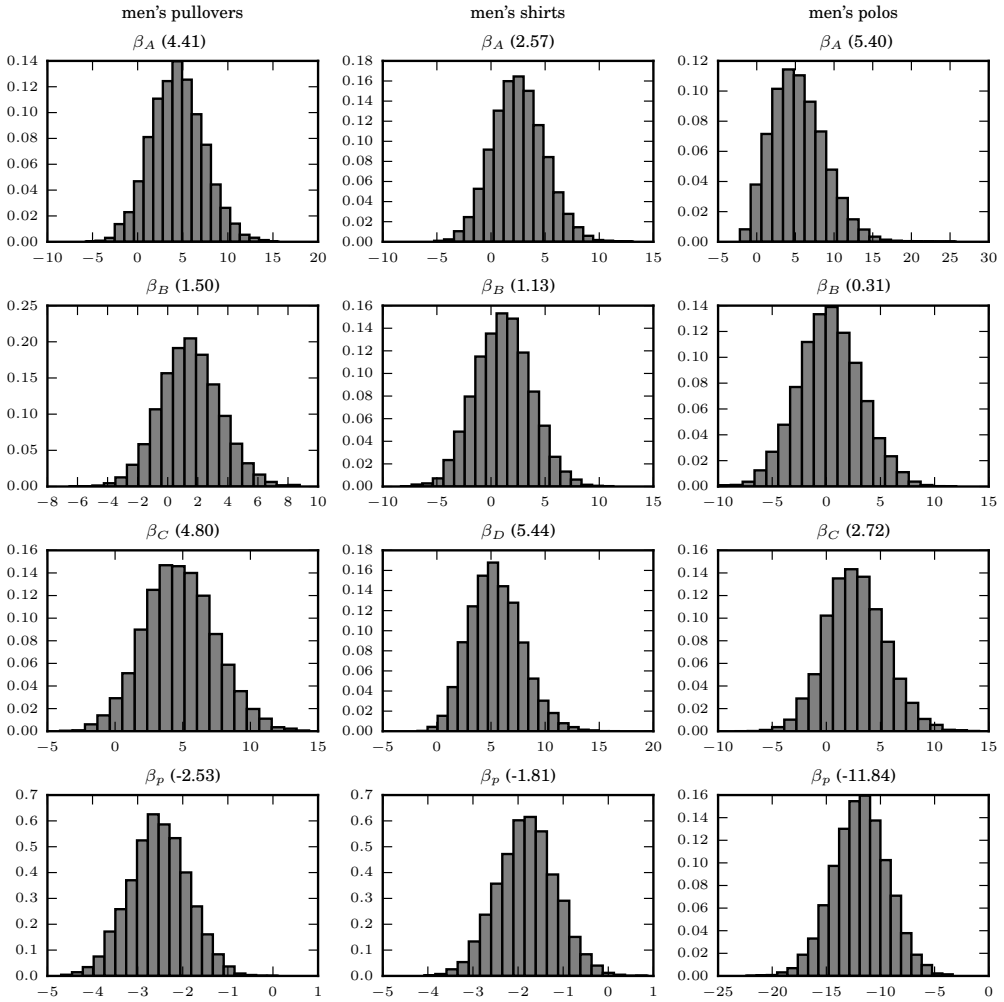


Figure 4.1: Posterior distributions for β_i of a specific customer. The columns pertain to products categories. The subscripts A, B, C, and D represent brand-specific constants and p denotes the price parameter. The corresponding means are in parenthesis.

Table 4.4: Number of customers that made a purchase in one, two or three categories for various sparsity factors.

Sparsity factor	0.0	0.2	0.4	0.6	0.8	1.0
# purchased in 1 category	0	1,600	3,200	5,600	8,800	12,000
# purchased in 2 categories	0	1,600	3,200	3,200	1,600	0
# purchased in 3 categories	12,000	8,800	5,600	3,200	1,600	0

4.5 Simulation Study on Data Sparsity

In this section, we numerically analyze the performance of the sampling scheme in (4.6)-(4.8) as the sparseness of the underlying dataset increases. To keep this simulation study in line with our empirical study in Section 4.4, we consider a setting with three product categories and four brands per category and $n = 12,000$. For each $i \in \{1, \ldots, n\}$, we sample β_i from the normal distribution with mean and covariance equal to the sample mean of μ and Σ in our empirical study in Section 4.4, which we denote by μ' and Σ', respectively. Then, we simulate six purchases per customer per category, where, for each purchase incidence, we randomly sample a product choice set with accompanying attributes from our dataset. This results in a dataset with similar characteristics as the real-life dataset used in Section 4.4.

To assess the consequences of increased sparsity of the data, we estimate the model multiple times, where each time the sparseness of the data increases. More precisely, in each estimation round, a subset of customers is selected, and all the observations from certain categories of these customers are permanently removed. Then, we apply our MCMC sampling scheme again to sample the parameter distributions and compare the mean of the covariances (i.e., the off-diagonal elements of Σ) to the "true" parameters contained in Σ'.

To quantify the sparseness of the data in each estimation round, we introduce a *sparsity factor*. The purpose and precise impact of this factor is presented in Table 4.4, in which it can be observed that the data sparseness increases as the sparsity factor increases from zero to one. A sparsity factor of zero corresponds to the case in which all customers make purchases in all categories and a factor of one means that each customer purchases in only one category.

In line with the empirical study, in each estimation round (i.e., for each sparsity factor), the first 20,000 draws are used as burn-in to allow the MCMC sampler to reach its stationary distribution. Then, the subsequent 20,000 draws are kept for inference on the parameters.

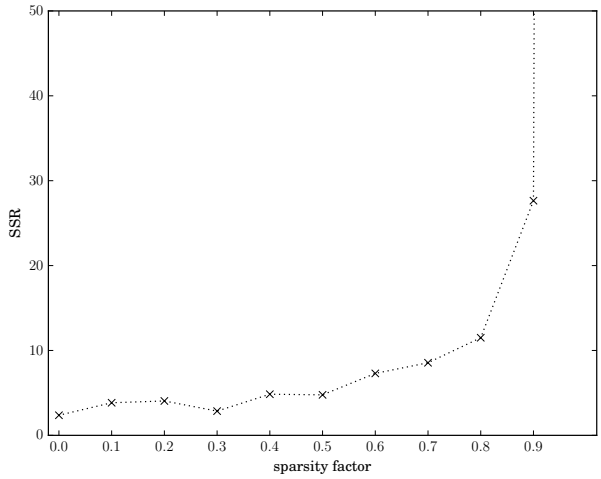


Figure 4.2: Sum of squared residuals (SSR) of the sampling average of Σ as the sparseness of the data increases.

In Figure 4.2, we present the quality of the mean parameter estimates, as the sparseness of the data increases. We do this by summing the squares of the differences between all covariance estimates and the true values, i.e., SSR $= \sum_i \sum_{j>i} \left(\hat{\Sigma}_{ij} - \Sigma'_{ij}\right)^2$, where $\hat{\Sigma}$ represents the mean of the samples of Σ. From the figure, we observe that the SSR worsens gradually from approximately 2 to 30, as the sparseness increases from 0.0 to 0.9. Eventually, as expected, at a sparsity factor of 1.0, the SSR increases dramatically, since some of the off-diagonal elements of Σ are no longer identified.

All in all, the results reveal that the estimation accuracy is fairly stable for a range of sparsity levels and that the estimation accuracy only deteriorates when a minor fraction of the customers buy in multiple categories.

4.6 Application: Promotion Optimization

The problem that we consider is to determine, for a given category, which product to offer to a customer and how deep to discount this product to maximize revenue. We assume that only a finite number of discount percentages is allowed. This resembles business practice (in which the number of different discount percentages is typically finite) and makes the optimization problem tractable.

Let $\mathcal{D}$ be the finite set of permissible discount fractions (e.g., $\mathcal{D} = \{0.0, 0.25\}$ means that either no discount or a 25% discount can be given) and, for each $c \in \{1, \ldots, m\}$, let $\mathcal{P}_c$

denote the assortment of products currently on stock in category c. With slight abuse of notation, for each $i \in \{1,\dots,n\}$, $c \in \{1,\dots,m\}$, $j \in \mathcal{P}_c$, and $s \in \{1,\dots,\zeta\}$, let $\beta_{icj}^{(s)}$ and $\beta_{icp}^{(s)}$ denote the s^{th} sample of the parameter pertaining to the brand of product j and the price sensitivity parameter, respectively, of customer i for category c. Let $c \in \{1,\dots,m\}$. Then, for each $i \in \{1,\dots,n\}$, we consider the following problem:

$$\max_{d \in \mathcal{D}, j \in \mathcal{P}_c} \underbrace{(1-d)\cdot \text{price}_j}_{\text{price of product } j} \cdot \underbrace{\frac{1}{\zeta}\sum_{s=1}^{\zeta} \frac{e^{\beta_{icj}^{(s)} + (1-d)\cdot \text{price}_j \cdot \beta_{icp}^{(s)}}}{e^{\beta_{icj}^{(s)} + (1-d)\cdot \text{price}_j \cdot \beta_{icp}^{(s)}} + \sum_{j' \neq j} e^{\beta_{icj'}^{(s)} + \text{price}_{j'} \cdot \beta_{icp}^{(s)}}}}_{\text{purchase probability of product } j} \tag{4.16}$$

Since both $\mathcal{D}$ and $\mathcal{P}_c$ are finite, we can solve the problem by enumerating the complete solution set of (4.16).

To illustrate the practical potential of our model, we set $\mathcal{D} = \{0.0, 0.25\}$ and set c as the category men's polos. We sample $\mathcal{P}_c$ randomly from the hold-out set for each $c \in \{1,\dots,m\}$ (cf. Section 4.4.1). Figure 4.3 illustrates for the category men's polos which customers received a discount of 25% (circle) and which did not receive a discount (triangle). Hence, the first pane illustrates for all customers for which it was optimal to offer a product of brand A, whether it was optimal to discount the product or not. The figure is based on 500 randomly selected customers to improve the readability of the scatter plot. We emphasize that the parameter that measures the preference for brand G was restricted to 0 for identifiability, hence the horizontal line.

All in all, Figure 4.3 indicates that it is optimal to discount if the brand preference is relatively low or the price sensitivity is relatively high or both. This can be seen from the fact that the circles (i.e., the customers for which a discount is optimal) can be approximately separated from the circles by drawing a decreasing line or curve in the plot.

A similar analysis for the categories pullovers and shirts proved to be much less profound than the results presented in Figure 4.3 for men's polos. Presumably, this is because the observed price sensitivity in these categories is relatively low. In turn, this can be attributed to the fact that our choice set construction method (see Section 4.4.2) may lead to choice sets that contain many cheap alternatives, which are in fact leftovers of odd-sized products that are not viable alternatives to many customers. For example: if a specific pullover in size XXXL was purchased heavily discounted at the end of the season, then we assume that this product was available on this day for the same price in all sizes. As a consequence, it is inferred that customers buy relatively expensive products, resulting in a biased estimate of the price sensitivity.

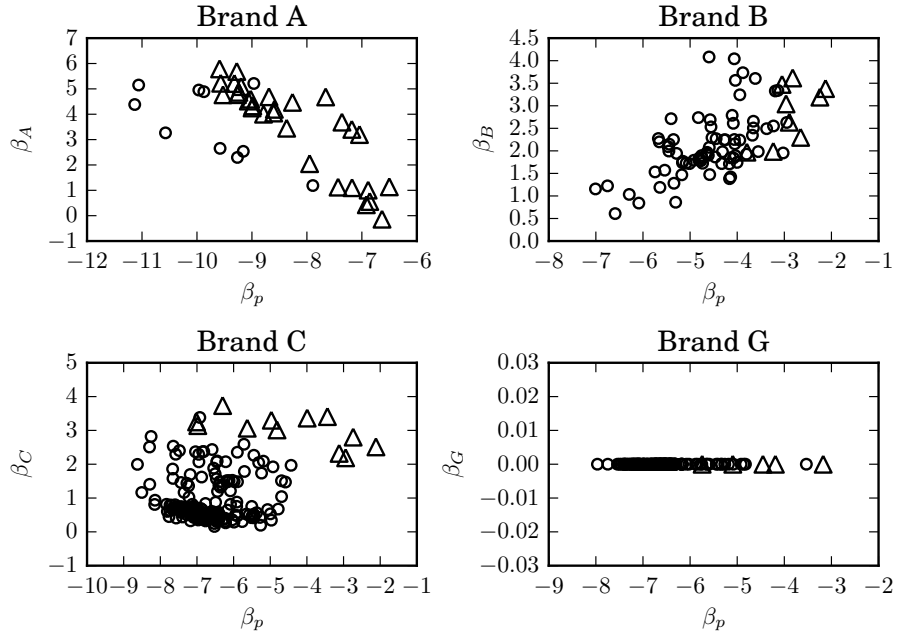


Figure 4.3: Each marker in the scatter plot pertains to a specific customer in the category men's polos. A circle implies that the customer received a discount, whereas a triangle indicates no discount. The x-axis and the y-axis pertain to the posterior mean of the brand preference and price sensitivity of the corresponding customer.

4.7 Conclusion and Future Research

The current work fits in a stream of research on multi-category customer choice modeling, which leveraged the increased availability of customer-level transaction data. Various decisions in a multi-category shopping setting have been considered. For example, many papers have studied the decision in which category or categories customers buy during a shopping trip (so-called category incidence). Others have considered the brand or product choice given a category incidence and the similarities in such choice behavior across categories. The current work fits in this last stream, and examines the similarities in decision making across categories of durable goods. In doing so, we examine how brand and price affect the product choice in the categories men's polos, men's shirts, and men's pullovers on a customer-level in a leading Dutch department store. We find that price sensitivity and brand preferences are similar across categories for the same customer. More precisely, the estimated correlation in price sensitivity parameters across categories is relatively high, as well as correlation in brand-specific parameters of the same brand across categories. This is consistent with existing literature on multi-category brand choice in grocery shopping. Our hierarchical Bayesian model does not require that all customers bought in all categories, which allows us to consider an application of our model to personalized promotions in categories in which customers (possibly) did not buy before.

There are several caveats and directions for future research to our study that we would like to discuss. First of all, since our model employs a full covariance matrix for the attribute weights, the model suffers from the curse of dimensionality and does not scale well to many categories. Even when very large transactional databases would allow us to identify possibly tens of thousands of covariances, this would be computationally very demanding and challenging to interpret from a practitioner's point of view. To resolve this, dimension reduction methods, such as developed by Singh et al. (2005), could be implemented. In addition, we believe that our choice set construction scheme leaves room for improvement. In the current setting, the choice set is inferred from the transactional dataset in a rather ad-hoc way, which may induce biases as discussed at the end of Section 4.6. To resolve this, the transactional data set should be enriched with data on (in-store) stock levels over time, since these reflect the true alternatives that were available when customers made their choice.

4.A Appendix: Gibbs Sampling of μ and Σ

Let $s \in \{2, \ldots, \zeta\}$ and recall that ν, V, $\overline{\mu}$, and σ are hyperparameters (see Section 4.3.5 for hyperparameter settings). The following follows from standard Bayesian theory on conjugacy (see, e.g., Rossi et al., 2012).

For sampling step (4.7), we sample $\Sigma^{(s)}$ from

$$\text{IW}(\nu + n, V + S),$$

where

$$S = \left(\beta_1^{(s)} - \tilde{\mu}, \ldots, \beta_n^{(s)} - \tilde{\mu}\right)\left(\beta_1^{(s)} - \tilde{\mu}, \ldots, \beta_n^{(s)} - \tilde{\mu}\right)^\top + \frac{1}{\sigma}\left(\tilde{\mu} - \bar{\mu}\right)\left(\tilde{\mu} - \bar{\mu}\right)^\top, \text{ and}$$

$$\tilde{\mu} = \frac{\sum_{i=1}^{n} \beta_i^{(s)} + \frac{1}{\sigma}\bar{\mu}}{n + \frac{1}{\sigma}}.$$

For sampling step (4.8), we sample $\mu^{(s)}$ from $N\left(\tilde{\mu}, \frac{1}{n+\frac{1}{\sigma}}\Sigma^{(s)}\right)$.

Chapter 5

Data-driven Consumer Debt Collection via Machine Learning and Approximate Dynamic Programming

5.1 Introduction

5.1.1 Background and motivation

In the U.S., $605 billion of household debt was delinquent as of March 31, 2018 (Federal Reserve Bank of New York, 2018). Of this amount, over $400 billion was delinquent for more than 90 days. For companies that rely on installment payments, this means it is of great importance to manage the collection of installments efficiently and collect as many payments as possible to assure business continuity and drive profitability. The potential, but also the complexity of doing so, was already recognized half a century ago by Cole (1968):

> "Collection work would be easier and the results better if there were some magic way in which each account could be immediately and accurately classified as to the reason for nonpayment and the collection method which would be most effective with that particular debtor. Sorting devices to perform such miracles unfortunately are not yet available, and until such become economically and mechanically feasible the responsibility for any classification, if made at all, rests with the credit personnel involved." (pp. 314-315).

With the increased availability of data and the development of sophisticated machine learning techniques, such "sorting devices" have now become reality.

In this chapter, we present a framework for the data-driven scheduling of outbound phone calls made by debt collectors. That is, we determine on a daily basis which debtors a debt collector should call to maximize the amount of delinquent debt recovered in the long term, under the constraint that only a limited number of phone calls can be made each day. These phone calls are used to persuade debtors to settle their debt, or to negotiate payment arrangements (e.g., a payment plan) in case debtors are willing, but unable to repay their debt. Scheduling these calls is challenging as it is difficult to assess the value of making a phone call to a debtor. This is because a priori the outcome of making a call is uncertain, and the extent to which a call attributes to a repayment is non-trivial. In general, the effect of phone calls on the repayment behavior of debtors depends on numerous interacting features, such as the time since the previous phone call, whether the debtor answered the call before, the amount of debt owed, the time of the month, and the persuasiveness of the agent who is calling. It is unclear what the effect of these (interacting) features is on the outcome of phone calls and, hence, on the effectiveness of a schedule of phone calls. This lack of structure and understanding drives our belief that a flexible non-parametric machine learning method would be most appropriate to leverage data for optimizing actions.

To this end, we show that the problem of scheduling phone calls is naturally formulated as a Markov decision process (MDP), but that a prohibitively large state space is required to capture the dynamics of the collection process appropriately. To alleviate this, we show how state-of-the-art machine learning methods can be used in an approximate dynamic programming (ADP) framework that is interpretable, highly scalable, and data-driven. We validate our proposed approach by means of a controlled field experiment with real debtors in a real business setting.

This research is carried out in collaboration with an anonymous debt collection agency from the Netherlands, to which we refer to as the Collector. The Collector provided the data required to estimate our models and implemented our methodology to conduct the controlled field experiment. The Collector handles about 250,000 collection cases each year, with a principal (monetary value) of approximately €120 million. Currently, the Collector schedules phone calls according to a static policy in which calls are scheduled based on a one-size-fits-all policy. Given that the Collector has carefully tracked all of its historical efforts and outcomes, we can leverage this data for the purpose of optimizing its collection process.

5.1.2 Contributions

To the best of our knowledge, this work is the first to incorporate modern machine learning methods into an ADP framework that is validated through a controlled field experiment in a real-life business setting. We take the problem of dynamically scheduling outbound calls for a debt collector—as naturally described by an MDP—and approximate state values using supervised machine learning. More precisely, we construct a binary classification problem to predict—based on a debtor's state—the likelihood with which a debtor is going to repay its debt. The debtor's state space is high dimensional and incorporates all static and dynamic information that characterizes a debtor at a given point in time. For the purpose of value function approximation, we multiply the likelihood with which a debtor settles its debt by the size of the debt—thereby obtaining an approximation for the expected value of a debtor given its current state. In doing so, we overcome the curse of dimensionality inherent to this problem by inferring the value of a debtor's state based on historical data in a highly scalable and flexible manner.

Based on our value function approximation, we compute for each debtor the *marginal value* of a phone call, which is defined as the change in the value function if we spend another phone call on this debtor. This leads to a particularly straightforward optimization procedure, namely, we prioritize the debtors that have the highest marginal value per phone call. The result is a policy that is interpretable (debtors with the highest marginal value on the effort are prioritized), highly scalable, and data-driven. In addition, the optimization procedure allows for straightforward implementation in business practice: arrivals of new debtors are naturally incorporated and an appropriate number of phone calls can be determined to be made on a given day, depending on the debt collector's capacity.

We validate our proposed methodology in a controlled field experiment conducted with real debtors. The results show that our optimized policy substantially outperforms the current scheduling policy that has been used in business practice for many years. Most importantly, our policy collects more debt in less time, whilst using substantially fewer resources—leading to a 47.2% increase in the amount of debt collected per phone call. We also identify a key managerial insight, namely that capacity is best spent on debtors that are more difficult to collect from. These are debtors that are in the collection process for a longer period of time, are less likely to pick up the phone, and have not partially repaid or promised to do so. These insights help managers better understand the dynamics of the debt collection process.

In summary, this chapter contributes to the existing literature on business analytics,

data-driven optimization, and that of ADPs in the following ways: i) we add to the debt collection optimization literature by presenting a novel, scalable, and flexible framework for daily data-driven scheduling of outbound calls; ii) we incorporate state-of-the-art machine learning methods to the ADP framework, which takes advantage of higher-order feature interactions and results in superior out-of-sample model fit for value function approximation compared to benchmark models; iii) we open the proverbial machine learning black box and identify generalizable insights for the improved scheduling of outbound debt collection phone calls; and iv) we validate our methodology by means of a controlled field experiment with real debtors.

The following section contains a review of existing literature on debt collection and debt collection optimization in particular. We discuss relevant literature that concerns approximate dynamic programming in Section 5.4, together with our proposed approximation method.

5.1.3 Literature review

Debt collection optimization

More than half a century ago, Mitchner and Peterson (1957) considered the problem of optimizing the collection of delinquent debt at Bank of America for various types of loans, such as car loans, personal loans, and real estate loans. They formulated the problem of collecting debt as an optimal stopping problem, in which the duration with which the collector should pursue the debtor was optimized, taking into account the cost of doing so. Their results show a potential increase in net profit of 33%.

Fifteen years later, Liebman (1972) developed a simple Markov decision process for optimizing credit control policies. They solve an example problem with four delinquency states, two amount owed states, two recent experience states, and three action strategies. However, the curse of dimensionality quickly becomes a significant challenge and no further progress on this topic was made until more recently by Abe et al. (2010), De Almeida Filho et al. (2010), and Miller et al. (2012).

In Abe et al. (2010) and the accompanying paper Miller et al. (2012), a framework for debt collection optimization is presented that, of the existing work, is closest to the approach considered in this chapter. In Abe et al. (2010) the collection process is modeled as a constrained MDP, which explicitly takes business, legal, and resource constraints into account. Subsequently, given the intractability of the MDP, a constrained Q-learning algorithm is proposed by means of which a policy can be obtained. In Miller

et al. (2012) the deployment of this methodology at the New York State Department of Taxation and Finance is described for which an increase in collected delinquent debt by 8 percent is reported over the first year, where an increase of 2-4 percent would otherwise have been projected.

Also from the operations domain, De Almeida Filho et al. (2010) present a study on the optimization of debt collection in the context of consumer lending. In their work, a dynamic programming approach is presented in which the monthly decision epochs pertain to deciding which action to take in the month to come. The value function corresponds to the future net discounted recovery rate and the transitions are assumed to be deterministic. Since the model assumes homogeneous debtors, the approach is especially useful to predict collection performance and resource requirements for aggregated portfolios of debtors for which it is reasonable to assume homogeneity. The authors refer to the importance and potential of tailoring the collection process to the individual debtor, but note that the data required for this purpose are hardly ever available in practice.

Credit scoring and valuation

In the field of finance, much research has been done on the credit-granting decision, i.e., whether to grant a loan to a potential new customer. Typically, the credit-granting decision for personal loans is made by means of credit scoring, which is a standardized method of assigning a score to potential customers that represent their creditworthiness—see Crook et al. (2007) for a literature review and Lessmann et al. (2015) for a benchmarking study on existing scoring models.

On the other hand, the valuation of existing credit—and existing unsecured credit in particular—is more closely related to our work since the debt that the Collector is trying to collect is essentially outstanding unsecured credit. Although much work has been done on the valuation of corporate credit and secured customer credit, the literature on unsecured consumer credit is sparse. The work of Chehrazi and Weber (2015) on dynamic valuation of delinquent credit card accounts models stochastic repayment behavior of individual debtors over time. They derive a self-exciting point process for repayment behavior and estimate the parameters of the process using the generalized method of moments. This model is then used to construct a dynamic collectability score to estimate the probability of collecting from a debt account, thus allowing for the valuation of credit card debt. In a subsequent paper, Chehrazi et al. (2018) formulates a stochastic optimal control problem from the self-exciting point process established in

Chehrazi and Weber (2015) and derives a semi-analytic solution. However, this solution was not analyzed empirically nor experimentally validated.

5.1.4 Outline

In Section 5.2, we provide an overview of how debt collection works and what the optimization problem is that debt collectors face. In Section 5.3 and Section 5.4, we formulate an MDP and provide an approximation for this MDP, respectively. Section 5.5 contains a description of the data used for model development and validation. In Section 5.6, we present the results of model estimation and validation. Section 5.7 presents the design and results of our controlled field experiment. We conclude in Section 5.8.

5.2 Problem Description

We first provide a high-level overview of the operations of a debt collector. Thereafter, we provide more details on the actual collection process. This is all based on the experiences of our industry partner (the Collector), but is illustrative for the debt collection industry in general.

5.2.1 High-level overview

In practice, a client that has overdue debt with a company is placed "in collections", which means that the debtor is transferred to either a specialized debt collection department within the company or to an external debt collecting agency that works on behalf of the company. In this work, we refer to both as a debt collector, i.e., a debt collector can be either the debt owner itself or a third-party debt collection agency working on behalf of the debt owner. The debtor typically incurs a collection fee that is added to the original debt to cover the additional costs of recovering the debt, and is regulated in many countries. In the problem that we consider, the collection fee is independent of the amount of debt owed and constant across debtors.

Once placed in collections, the debt collector pursues the debtor to settle the debt plus the collection fee by sending out letters and e-mails, and through phone calls made by its agents. Amongst the Collector's clients are utility providers, credit facilitators, and health care providers, which operate in the business-to-consumer market.

The process of the Collector comprises two phases. Upon arrival, a debtor first enters the collection phase, in which the Collector pursues the debtor to repay the debt plus the collection fee through letters, e-mails, and phone calls. During this phase, the collector acts cooperatively towards the debtor, and can offer payment plans if debtors are willing, but not able to pay on a short-term notice. As such, this phase can take from a few days (in case the debtor pays immediately) up to a few months (in case the debtor does not pay at all or gets involved in a payment plan).

When the Collector is unsuccessful in recovering the debt during the collection phase, it chooses to either write off the debt or invoke a legal procedure. The former happens when, for example, the debtor is deceased or has declared bankruptcy. The latter means that a bailiff is invoked, who will send out a subpoena and ultimately can confiscate property if necessary. Whether a debtor is escalated to the legal phase or written off is determined case by case and depends, amongst other things, on the amount of outstanding debt and the likelihood of recovering the debt through legal procedures. Since this phase requires legal assessment by an expert, it is very expensive and the outcome is highly uncertain. Hence, recovering debt before the legal phase is deemed beneficial for both collector and debtor. As such, the legal phase is excluded from the optimization procedures proposed in this work and our objective is to maximize recovered debt during the collection phase, which is described in greater detail in the following section.

5.2.2 Collection phase

The collection phase is characterized by four sequential letters (sent via both post and e-mail simultaneously), where each letter has a seven-day payment notice and communicates with increasing urgency the necessity to repay the debt. The letters are sent between seven to ten days of each other. The fourth and final letter communicates the severe (financial) consequences of the legal procedure that is possibly invoked if the debtor does not settle.

In between the letters (or after the final letter), the Collector is free to call debtors at its discretion. This is considered a vital tool during the collection process, as the phone calls allow the agents to inform the debtor about the situation along with the consequences of non-payment, and to make an assessment of whether the debtor is willing and/or able to pay. Figure 5.1 provides a schematic illustration of the collection process.

The optimization problem that the Collector encounters, is deciding each day which debtors should be called to maximize recovered debt, given the finite and inflexible capacity of its workforce. In practice, this implies that the Collector has to decide on

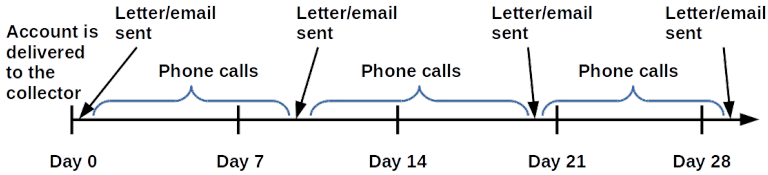


Figure 5.1: The standard operating timeline of a debt collection agency.

a prioritization on the debtor portfolio that indicates which debtors should be called first. Currently, the Collector's policy is to schedule a phone call each time a debtor has received a new letter. In addition, if a debtor agreed on a payment plan and failed to comply with its conditions, a call is scheduled as well. In case capacity is insufficient, the Collector's managerial staff makes an assessment of which debtors should be called first. Given the labor-intensive nature of the phone calls, the gains from optimizing the prioritization of calls are potentially substantial.

5.3 Model Description

The problem of optimizing debt collection efforts over time in the current context is formulated as an MDP with an infinite time horizon and decision epochs in discrete time. This suits the approach of the Collector, since in principle the Collector operates indefinitely and decisions are made at discrete points in time (i.e., daily). To formalize the MDP, we assume that at any given point in time, the Collector has at most $N \in \mathbb{N}$ debtors in its portfolio. Practically, this means we set N arbitrarily large such that the Collector never has more than N debtors in its portfolio.

5.3.1 State space

We denote the state space of each of the (at most) N debtors by $\mathcal{X}$ and the state space of the portfolio of debtors by $\bar{\mathcal{X}} := \mathcal{X}^N$ (i.e., the N-fold Cartesian product of $\mathcal{X}$). In our formulation, each of the N parts of the state space is utilized by different debtors over time—each part $\mathcal{X}$ of the state space $\bar{\mathcal{X}}$ functions as a slot for storing the information of one of the debtors. A slot becomes available for new arriving debtors once efforts on the existing debtor are terminated because the debt is recovered or written-off.

The state space below is chosen to accommodate for the data to which we apply our methodology (as described in Section 5.5). We divide the debtor state space $\mathcal{X}$ into

debtor-specific features, historical-interaction features, and seasonalities as follows. Here, [B], [I], [N], and [C] indicate whether it is a binary, integer, numerical, or categorical variable, respectively.

Debtor specific: 1) initial debt amount [N], 2) customer tenure [B], 3) has partially repaid debt [B], 4) repayment plan in place [B], 5) phone number is available [B], 6) e-mail address is available [B], 7) product type [C], 8) amount repaid already [N], 9) Collector collected from debtor before [B], 10) average income in the postal code area of the debtor [N], 11) share of people under 30 in postal code area of the debtor [N], 12) current substatus [C], 13) passed final letter [B].

Here, 2) indicates when the debtor became a customer of the debt owner: the exact time was not provided, instead we have an integer that represents the inverse order in which the debtor became a customer relative to all customers of the debt owner (the larger the value means the debtor was a customer of the debt owner for a longer period of time); 5) pertains to whether the debt owner provided the Collector with a phone number of the debtor. If this is not the case, the Collector may still be able to call the debtor by searching manually in publicly available resources for potential phone numbers that match to the name and address of the debtor; 7) refers to the product or service that the debtor purchased and led to the debt; 12) refers to a debtor's state description used internally by the Collector to characterize a debtor at a given point in time; 13) refers to whether the debtors have received the final (i.e., fourth) letter.

Historical interaction: 1) has answered a phone call [B], 2) promised to repay [B], 3) number of previous collector-debtor interactions [B], 4) number of previous phone calls [B], 5) days since promise to repay [B], 6) days since last collector-debtor interaction [B], 7) days since last phone call [B], 8) days since last answered phone call [B], 9) days since last incoming contact [B], 10) days since last incoming e-mail [B], 11) days since last incoming phone call [B].

Here, 2) and 5) refer to the event in which the debtor has (verbally) promised the Collector to settle the debt; in 3) and 6) the word 'interaction' includes both collector- and debtor-initiated communication efforts, and also phone calls that did not get through count as an interaction—thereby using it in a broader sense than usual.

Seasonality: day of week [C], week of month [C].

For features where missing values are possible, a unique integer is used as replacement for missing values. An example of this is the feature *days since last phone call* for cases

where no phone calls have previously been made to the debtor.

5.3.2 Action space and value function

Regarding the action space, for a given day let $a \in \{0,1\}^N$ describe which debtors will be called: $a_i = 1$ for $i \in \{1,2,\ldots,N\}$ indicates that a call is made to debtor i, and $a_i = 0$ means no call is made on a given day. In some cases, it is undesirable to call a debtor (e.g., when the debt is currently being further investigated because the debtor disputed the debt). Therefore, we construct the action space as follows. Let $i \in \{1,\ldots,N\}$, $x = (x_1,\ldots,x_N) \in \bar{\mathcal{X}}$ with $x_i \in \mathcal{X}$, and let $\mathcal{A}'(x_i)$ denote the action space pertaining to debtor i, so that $\mathcal{A}'(x_i)$ equals $\{0\}$ if no call is allowed and $\{0,1\}$ when a call to debtor i is allowed. In addition, let $c_t \in \mathbb{N}$ denote the (deterministic) capacity on day t, i.e., the maximum number of phone calls that can be made on day t, where t counts the number of days since the collection process was initiated. Accordingly, we define

$$\mathcal{A}_t(x) := \left\{ (a_1,\ldots,a_N) \,:\, a_i \in \mathcal{A}'(x_i),\ i = 1,\ldots,N,\ \sum_{i=1}^{N} a_i \leq c_t \right\}$$

as the action space on day t. In case slot i of the state space is not used, $\mathcal{A}'(x_i) = \{0\}$ for all $x_i \in \mathcal{X}$.

Furthermore, on day t, for $x, y \in \bar{\mathcal{X}}$ and $a \in \mathcal{A}_t(x)$, let $p(x,a,y)$ denote the probability of moving from state x on day t to state y on day $t+1$, when choosing action a and let $r(x,a,y)$ denote the amount of debt recovered (i.e., repaid and received) when moving from state x to state y choosing action a. The possible arrival of new debtors is implicitly incorporated in $p(x,a,y)$. Then, the optimality equation becomes

$$V_t(x) = \max_{a \in \mathcal{A}_t(x)} \sum_{y \in \bar{\mathcal{X}}} p(x,a,y)\left(r(x,a,y) + \gamma V_{t+1}(y)\right), \tag{5.1}$$

for $t = 0,1,2,\ldots$, where $V_t(x)$ denotes the total expected discounted reward when being in state $x \in \bar{\mathcal{X}}$ at day t, and $\gamma \in (0,1)$ denotes an appropriate discount rate. The function $V_t : \bar{\mathcal{X}} \to \mathbb{R}$ is often referred to as the *value function*.

Since the state space $\bar{\mathcal{X}}$ consists of all debtor information, the formulated MDP has a high-dimensional state space. Moreover, parts of the state space are unbounded (e.g., the number of collector-debtor interactions). This makes it intractable to solve the MDP even numerically. The MDP, however, has structural properties that facilitate the computation of near-optimal policies. First, the debtors in the portfolio behave independently of each other, i.e., changes to the state and repayment probability of one debtor

do not affect the repayment probability of the other debtors. Second, the dependence in the problem formulation is only due to the capacity constraint c_t on day t. Hence, a natural approximation that breaks the dependence arises when the Collector solves a stochastic knapsack problem based on the state of the debtors in the portfolio on that day. The knapsack has size c_t on day t, and the expected value of each item in the knapsack will be given by the expected gain in the value function from calling the debtor. Note that in this formulation, the discount factor naturally disappears since future arrivals do not affect current decisions. In the next section, we elaborate on how to estimate the value function of each debtor.

5.4 Value Function Approximation with Machine Learning

We use value function approximation (VFA) to approximate the value of the states of the MDP described in the previous section. Any function can be used to approximate the value function, including radial basis functions, polynomials, neural networks, and decision trees (Bertsekas and Tsitsiklis, 1995). VFA has been successfully applied in optimization in a variety of problems, such as large-scale resource allocation (Powell and Topaloglu, 2006), multi-priority patient scheduling (Patrick et al., 2008), and autonomous inverted helicopter flight (Ng et al., 2006). Recent breakthroughs in machine learning—notably convolutional neural networks—have sparked the field of deep reinforcement learning, which allows for VFA through visual images. For example, AlphaGo was able to exploit this approach by successfully approximating the 10^{170} state space in the game of Go and defeat the world's best human players (Silver et al., 2016).

In this chapter we use another state-of-the-art machine learning algorithm for VFA, namely, gradient boosted decision trees (GBDT). This is a more suitable algorithm for prediction problems that are arranged in the standard tabular structure and has been the dominant algorithm in winning well over half of all machine learning competitions in 2015, including the KDD Cup (Chen and Guestrin, 2016). It was also found by Olson et al. (2018) to be the best algorithm when benchmarked against twelve other algorithms for 165 publicly available classification problems. In Section 5.4.3, we provide details on the GBDT algorithm.

We use the GBDT model to construct a mapping $\hat{V} : \bar{\mathcal{X}} \to \mathbb{R}$ that approximates the value function, thereby circumventing the problem of having to solve (5.1). This approximation is used to optimize the actions, i.e., to determine which debtors are to be called on a given day. In the following two sections, we show how we construct the mapping $\hat{V}$ (Section 5.4.1) and optimize actions based on this approximation (Section 5.4.2).

5.4.1 Estimating the predicted repayment probability

To approximate the value of a debtor being in a particular state, we estimate the debtor's predicted repayment probability (PRP), which is defined as the likelihood of recovering the full debt during the collection phase. Partially repaid cases are considered to be unpaid as the Collector only receives credit for fully collected cases. Our approach is to estimate the PRP based on historical data by means of a GBDT model as follows. Suppose a certain debtor is $k \in \mathbb{N}$ days into the collection process, and consider all closed cases that once were k days into the collection process as well, i.e., all closed cases that either did not settle their debt within k days or were not written off within k days. We use these closed cases to train a GBDT model that predicts the likelihood of recovering the debt of the debtor currently considered. We formalize this procedure as follows.

Let $n \in \mathbb{N}$ be the total number of closed cases in our dataset, i.e., cases for which the debt was either recovered or written off, and for which the debtor is no longer being contacted. Let $i \in \{1, 2, \ldots, n\}$ and define $\tau_i \in \mathbb{N}$ as the total number of days debtor i spent in the collection process. For all $s \in \{1, 2, \ldots, \tau_i\}$, let $x_i^{(s)} \in \mathcal{X}$ be the state of debtor i at s days since arrival. We optimize the phone calls during the first $K \in \mathbb{N}$ days of the collection process of each debtor. Although, theoretically, K is unbounded, in our practical implementation we set K such that virtually all calling efforts take place in the first K days. For all $k \in \{1, \ldots, K\}$, denote by

$$\mathcal{I}_k := \{i : k \leq \tau_i,\ i \in \{1, \ldots, n\}\}$$

the index set containing all the closed cases in the dataset that were still in the collection process k days after arrival.

Furthermore, we denote by $y_i \in \{0, 1\}$ the eventual outcome of the collection process: $y_i = 1$ if the debt of debtor i was fully recovered after τ_i days, i.e., during the collection phase, and $y_i = 0$ otherwise, meaning that the debt was either written off or recovered after legal actions. Hence, $x_i^{(k)}$ and y_i are the state of debtor i after k days and the eventual outcome of the collection process, respectively, for all $k \in \{1, \ldots, K\}$ and all $i \in \mathcal{I}_k$.

Our approach is to train one GBDT model for each number of days since arrival $k \in \{1, \ldots, K\}$ as follows. Let $k \in \{1, \ldots, K\}$. Then, we train model k by using $(x_i^{(k)})_{i \in \mathcal{I}_k}$ as features (or independent variables) and $(y_i)_{i \in \mathcal{I}_k}$ as target (or dependent) variables. We denote the trained GBDT model by $f_k : \mathcal{X} \to (0, 1)$, where f_k maps the state of a debtor after k days to a prediction for the likelihood that the debt is eventually recovered. This

likelihood is exactly the PRP that we introduced earlier on, i.e., if debtor $i \in \mathcal{I}_k$ is in state $x \in \mathcal{X}$ after k days, then $f_k(x)$ represents its PRP.

We train a single model for each number of days since arrival because the data is unbalanced in the sense that there are many more observations for debtors that are earlier in the collection process (i.e., $|\mathcal{I}_k| \geq |\mathcal{I}_{k+1}|$ for each $k \in \{1, \ldots, K-1\}$). This is because cases are closed as soon as the debt is fully recovered or written off. If we train a single model, this could cause the GBDT model to be biased toward better predicting the early part of the process at the expense of the later part. To alleviate this, we follow the aforementioned approach in which we split the data by days after arrival into K sets and train K models.

Summarizing, we compute the PRP of a debtor on a given day by considering debtors that once were in a similar situation before, given that the days since arrival is highly correlated with the rest of the collection process.

5.4.2 Approximating the value function

To approximate the value of the state of a particular debtor, we multiply the debtor's PRP by its outstanding debt. More precisely, let $x = (x_1, \ldots, x_N) \in \bar{\mathcal{X}}$ be the state of the debtor portfolio at a certain point in time and let $k_i \in \mathbb{N}$ denote the number of days debtor $i \in \{1, \ldots, N\}$ has been in the collection process. Our approximation for the value of being in state x is

$$\hat{V}(x) := \sum_{i=1}^{N} f_{k_i}(x_i) \cdot \text{debt}_i, \tag{5.2}$$

where debt_i denotes debtor i's current outstanding debt. When slot $i \in \{1, \ldots, N\}$ of the state space is not used, we set $\text{debt}_i = 0$. Observe that, when the objective is to maximize the number of fully collected cases (irrespective of the amount of debt recovered), we can accommodate for this by setting $\text{debt}_i = 1$ for all $i \in \{1, \ldots, N\}$.

Our proposed approximation in Equation (5.2) implies that we consider the Collector's portfolio on a particular day as an assortment of independent debtors in different states of the collection process, and compute the value of the portfolio as a sum of their individual values. This approximation allows us to evaluate policies by computing the difference in PRP with and without making a phone call to a particular debtor.

To formalize this, let $\psi : \mathcal{X} \to \mathcal{X}$ be the mapping that takes as input a debtor's state and then updates this state as follows: i) increase the feature *number of previous collector-*

debtor interactions by one; ii) increase the feature *number of previous phone calls* by one; iii) set the feature *days since last collector-debtor interaction* to zero; and iv) set the feature *days since last phone call* to zero (see also the state space description in Section 5.3.1). Our approach is to determine the marginal value of making an additional call to debtor $i \in \{1, \ldots, N\}$ by computing

$$\left[f_{k_i}(\psi(x_i)) - f_{k_i}(x_i)\right] \cdot \text{debt}_i. \tag{5.3}$$

Recall that f_{k_i} maps debtor i's state to the PRP, i.e., to a prediction of the likelihood that debtor i will *eventually* repay, without needing to explicitly consider potential future states. Hence, Equation (5.3) provides us with a measure to compare the added value of calling different debtors. Naturally, the policy on day t is to call the c_t debtors for which Equation (5.3) is the highest (recall that c_t denotes the capacity of the Collector on day t). In the following section, we provide background on GBDT and discuss why it works well in this particular case.

5.4.3 Gradient boosted decision trees

GBDT, also called *gradient boosting machines* and *multiple additive regression trees*, falls under the general paradigm of ensemble methods in machine learning (Dietterich, 2000). The algorithm works by constructing multiple decision trees using the *classification and regression trees algorithm* (CART, Breiman et al., 1984) and combining these into a so-called committee, in which the predictions of the individual trees are combined to form one prediction (usually via a weighted average). We first describe how CART works and what its drawbacks are. Then, we explain how ensembles of trees overcome these drawbacks. Finally, we describe the GBDT algorithm and discuss why it works for our problem of predicting the repayment of debt.

The CART algorithm works by recursively partitioning the feature space into nonoverlapping rectangular subsets and making a prediction for the target variable for each of these subsets. This is done by splitting, in each recursion, the feature that minimizes a certain error metric (e.g., mean squared error or Gini impurity). This procedure is myopic in a sense that the partitioning decision does not consider future partitionings. As a result, CART does not guarantee a globally optimal partitioning.

A major drawback of CART is its propensity to overfit on training data, which results in a model that generalizes poorly to unseen data. Ensembles of CART models have been successfully used to overcome this. Early ensembling techniques, such as bootstrapped aggregating, commonly referred to as *bagging*, work by generating multiple versions

of a prediction algorithm by using randomly selected subsamples of the training data (Breiman, 1996). The random forest algorithm is an example of a bagging algorithm. Subsampling observations via bootstrapping adds variation to the training data, which leads to significantly different trees being built, resulting in reductions in error rate by 20-89% (Breiman, 2001).

Unlike bagging, where trees are built independently, GBDT builds trees sequentially. This is called *boosting* and works as follows. The goal of GBDT is to minimize a loss (or: objective) function that maps the predictions to a score that measures the quality of the predictions. Theoretically, any differentiable function can be used as a loss function. We use the *logarithmic loss* function, which is the standard choice for binary classification problems and is defined as follows. Suppose we are training model f_k for $k \in \{1, \ldots, K\}$, then the logarithmic loss function $L : \mathbb{R}^{|\mathcal{I}_k|} \to \mathbb{R}$ is

$$L(z) := -\left(\sum_{i \in \mathcal{I}_k} y_i \cdot \log(\sigma(z_i)) + (1 - y_i) \cdot \log(1 - \sigma(z_i)) \right), \tag{5.4}$$

where $\sigma : \mathbb{R} \to (0, 1)$ is defined by $\sigma(u) := (1 + e^{-u})^{-1}$ for $u \in \mathbb{R}$. The GBDT algorithm repeats Step 1-3 below a prespecified number of times, where $\epsilon > 0$ is set as a hyperparameter:

Step 0. Initalize with $z_i \leftarrow \sigma^{-1}\left(\frac{1}{|\mathcal{I}_k|} \sum_{j \in \mathcal{I}_k} y_j\right)$ for all $i \in \mathcal{I}_k$.

Step 1. Compute the gradient of the loss function $\frac{\partial L(z)}{\partial z_i} = \sigma(z_i) - y_i$ for all $i \in \mathcal{I}_k$.

Step 2. Train a regression tree using $-(\sigma(z_i) - y_i)$ for all $i \in \mathcal{I}_k$ as the target variables.

Step 3. Update $z \leftarrow z + \epsilon z'$, where $z' \in \mathbb{R}^{|\mathcal{I}_k|}$ are the predictions from Step 2. Go to Step 1.

When the algorithm terminates, $\sigma(z_i)$ is GBDT's prediction for y_i for all $i \in \mathcal{I}_k$.

By iteratively building regression trees on the negative gradient in Step 2, newly built trees are optimized for observations that are difficult to predict, thereby improving the overall model fit with each iteration. For a more detailed discussion on GBDTs we refer the reader to Friedman (2001) and Friedman et al. (2001).

Improving the model fit of the training data does not guarantee generalization to unseen data. Therefore, a cap in the number of iterations is required to prevent overfitting. The cap in the number of iterations, along with other hyper-parameters, such as maximum depth per tree, can be tuned using a training-validation framework. The implementation of GBDT used in this chapter is LightGBM, which is a fast and distributed open

Table 5.1: Observable collector-debtor interactions.

Variable	Type	Description
Debtor ID	Integer	-
Date	Date	-
Communication type	Categorical	Letter, e-mail, or phone call
Communication direction	Binary	In- or outbound
Reached	Binary	In case of outbound phone call
Document type	Categorical	In case of outbound letter / e-mail
Promised to pay	Binary	If debtor promised to pay

source GBDT framework developed by Microsoft (Ke et al., 2017).

CART, and GBDT, in particular, is well suited for our prediction problem for two reasons. First, since CART works by partitioning the data, it is invariant to monotonic transformations of the features. This differs from models such as logistic regression where substantial efforts in finding the best functional transformations of the features are required to tune the model to achieve better prediction performance. This means that we can directly use the debtor's collection state as features in a CART model without performing any functional transformations. Second, as a consequence of recursive partitioning, CART implicitly takes into account feature interactions that can lead to improved prediction accuracy and better state-value approximations. Again, for other models such as logistic regression, the feature interactions must be defined manually.

5.5 Data Description

To train and validate our proposed value function approximation method, described in the previous section, we rely on a dataset provided by the Collector. This dataset contains information on 80,138 debtors that arrived between January 1, 2014 and September 30, 2016. All these debtors are individuals, who are clients of the same insurance company. This insurance company offers all kinds of insurance products, such as car and travel insurance plans. The dataset comprises four data sources: i) debtor-specific information: customer tenure, the type of insurance product, whether the Collector has tried to collect from the debtor on a previous occasion, date of arrival, postal code, original debt, and the collection fee; ii) log of historical interactions between the Collector and the debtor, see Table 5.1; iii) log of incoming payments; and iv) log of status and substatus changes. The status and substatus changes pertain to information that is used by the Collector to characterize the current state of a debtor. The status is active (the debtor is

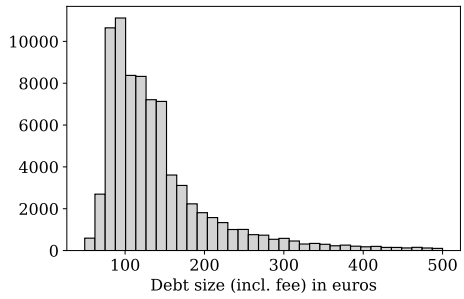


Figure 5.2: Distribution of debt amounts. For readability, we capped the debt at €500.

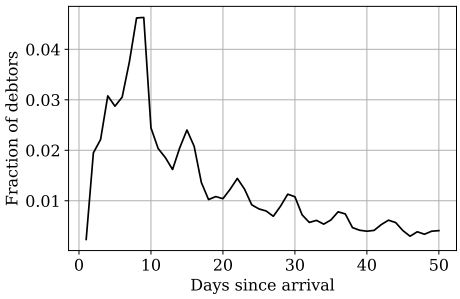


Figure 5.3: Full repayment over time. The y-axis pertains to the fraction of the debtors that repaid their debt (in full) on the corresponding day.

currently being pursued), inactive (the debt has been paid or the debt has been written off), or on hold (the case is currently being investigated, i.e., there is reason to believe that the debt has already been paid or is inadmissible). The substatus describes the status in greater detail—it indicates: in which stage of the process a debtor is (i.e., which document has been sent most recently); if the debtor has agreed to a payment plan; if the debtor has violated a payment agreement; if the contact details are incorrect; etc. In addition, we enriched the dataset by adding for each debtor the average disposable income in the postal code area where the debtor resides plus data on the distribution of age groups by leveraging publicly available data[1]. Using these data sources we are able to compile the state space features described in Section 5.3.1.

In Figure 5.2 to Figure 5.5, we illustrate the characteristics of the data that we used. In Figure 5.2, we present a histogram of the initial debt amount including collection fee (2,430 debtors, or 3%, have a higher initial debt than €500, with a maximum of €4,779). The histogram reveals that most debts are in the €50–€200 range, but that the distribution has a heavy right tail, with occasionally large amounts. It holds that the smaller the debt, the greater the likelihood that the debt is recovered by the Collector—of all amounts smaller than €300, for example, approximately 72% is recovered in the first fifty days, whereas for amounts larger than €300 this is only 53%.

Figure 5.3 illustrates the fraction (or relative frequency) of repaying (the full amount) during each of the first fifty days of the collection process. For example, approximately one percent of all debtors pay off all of its remaining debt on the twentieth day since arrival. The figure reveals that after the first week there is a negative trend observable,

[1]Central Bureau of Statistics, https://www.cbs.nl/nl-nl/maatwerk/2017/15/besteedbaar-inkomen-per-postcodegebied-2004-2014, accessed October 11, 2017.

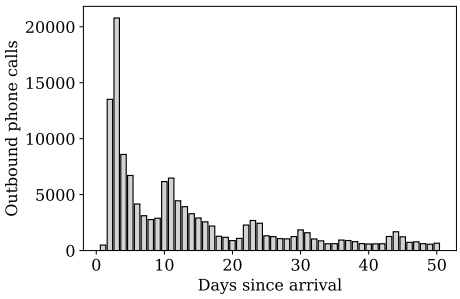


Figure 5.4: Number of outbound calls.

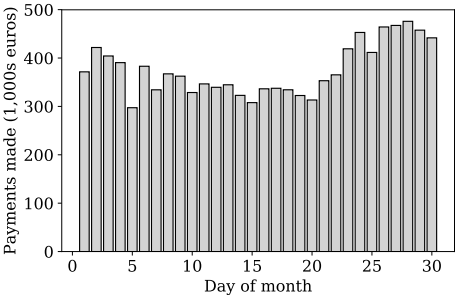


Figure 5.5: Debt recovered for each day of the month.

but that jumps occur regularly, which are due to letters that are sent and payment due dates that expire.

The bar chart in Figure 5.4 illustrates the number of outbound calls made on each of the first fifty days of the collection process of all debtors. For example, approximately 20,000 calls are made to debtors on the third day after their arrival. The figure clearly shows that the calls are clustered after letters have been sent—around day 2, 11, 21, and 30. The last cluster that is observable, around day 43, corresponds to the payment due date of the fourth and final letter.

Finally, in the bar chart in Figure 5.5 the inflow of money over different days of the month is illustrated (e.g., the first bar corresponds to the amount of debt that is paid on the first day of the month). The inflow of money peaks at the end of the month after people have received their paycheck, and then gradually decreases again over time.

5.6 Model Estimation Results

In this section, we describe how we train and validate the GBDT binary classification models that predict if debtors are going to repay, based on their current state (see Section 5.4.1). First, we set up a training and validation framework (Section 5.6.1). Then, we evaluate the prediction performance (Section 5.6.2) and illustrate debtor-specific prediction trajectories (Section 5.6.3). Section 5.6.4 contains an analysis on feature importance. Finally, in Section 5.6.5, we analyze the marginal effect of phone calls on PRP.

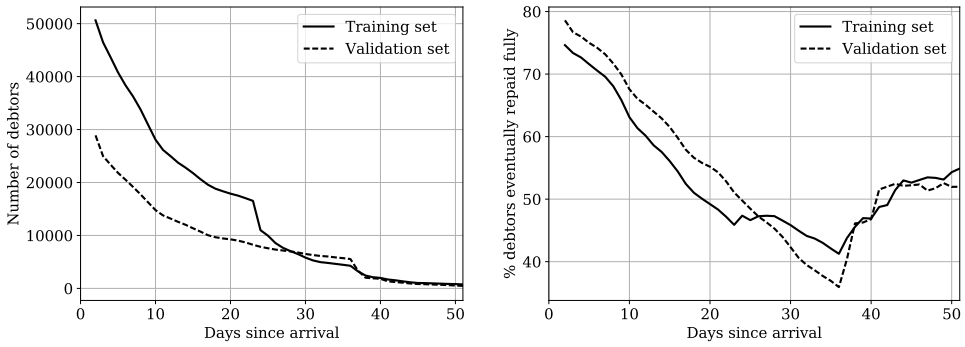
5.6.1 Training and validation data

We split the dataset into a training and validation set: debtors who arrived between January 1, 2014 and July 19, 2015 are used for training, and debtors who arrived between July 20, 2015 and September 30, 2016 are held out from training and used for validation.[2] The decision to split the training and validation set by date is to mimic practice, where only data of the present is available when making predictions about the future. We chose the specific split date arbitrarily, but in practice the split is often made such that 60-70% of the data is used for training and the remaining 30-40% is used for validation. As a result, the training and validation set contain 50,624 and 28,900 debtors, respectively. A small number of debtors had only inadmissible statuses and therefore were not included for training or evaluation, hence the total number of debtors between the training and validation set is less than the total reported in Section 5.5.

Figure 5.6 (a) shows the number of debtors still in the collection process as time passes for both the training and validation datasets. The figure reveals that the number of debtors decreases as time goes by, which is the result of repayment, writing-off debt, and legal action. There is a sudden drop in the number of debtors around day 23 and day 37 in the training and validation set, respectively. This is due to changes in the collection process on January 1, 2015, where the date of initializing the legal phase was postponed from day 23 to a variable number between days 32 and 42.

Figure 5.6 (b) illustrates the relative frequency of debtors for various days since arrival from the training and validation sets. For a specific number of days since arrival $k \in \{1, \ldots, 50\}$, the graph illustrates the percentage of debtors that eventually repaid during the collection process given that the debtor is still in the collection process after k days. The percentage of debtors that settles its debt completely initially decreases as their cases are further into the collection process. This suggests that debtors who are more able and willing to repay their debt will do so quickly, while those who are less able or willing to repay may require more effort. Around day 35, the percentage starts increasing again, which can be attributed to the fact that the Collector starts writing-off debtors that are impossible to collect from and, consequently, the remaining debtors are not written-off and are more likely to repay.

[2]There is often ambiguity between the definition of validation and test sets. In this work, we define the validation dataset as the holdout set which we use to evaluate our GBDT models.



(a) (b)

Figure 5.6: Number of debtors (a) and percentage of debtors that repaid (b) for various days since arrival.

5.6.2 Debt repayment prediction performance

We measure the quality of the GBDT binary classification models $f_1, \ldots, f_K$ by their ability to distinguish repaying from non-repaying debtors. Recall that, for all $k \in \{1, \ldots, K\}$, f_k maps the state of a debtor after k days to a prediction for the probability that this debtor repays its debt prior to legal action (i.e., the PRP, see Section 5.4). Due to the low number of debtors remaining late in the collection process, we limit our model to only consider debtors up to 50 days since arrival (where only 1,147 and 620 debtors remain in the training and validation set, respectively), thus we set $K = 50$. Finally, the first phone calls were made starting from day 2 so no debtors are considered for day 1.

The models $f_2, \ldots, f_{50}$ are trained using the training data as described in Section 5.6.1. For details on the construction of the features (independent variables) and the target (dependent) variable for each model, we refer to Section 5.4.1 and Section 5.4.2. Using the trained models, for each debtor in the validation set we compute the PRP for each day this debtor was in the collection process. To compare the PRPs with the actual outcomes, we use the area under the receiving operator curve (AUC). Here, the AUC can be interpreted as the probability that we rank a randomly selected debtor that eventually settles its debt as more likely to repay than a randomly chosen debtor that did not repay (Fawcett, 2006). We achieve an AUC of 0.689 where, in comparison, the AUC score of random guessing (i.e., predicting 0 or 1 with equal probability) or naively setting the repayment probability equal to the empirical probability is equal to 0.5.

We use AUC since it measures how well we are able to rank debtors based on their likelihood of repayment, which fits our optimization procedure in which we rank the

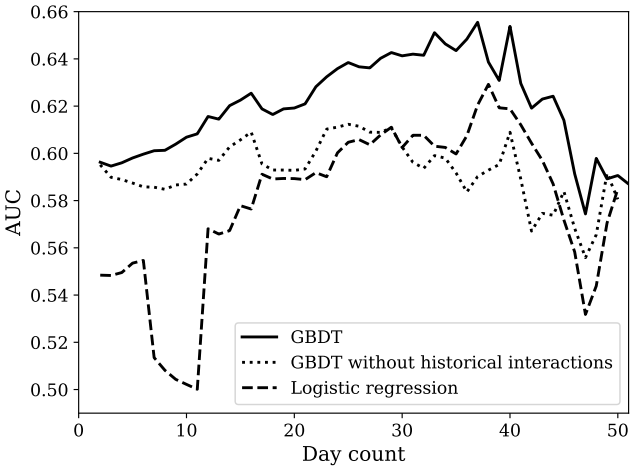


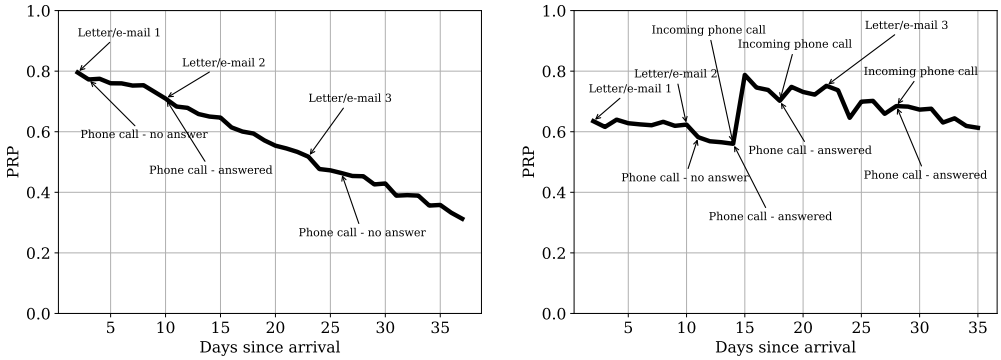
Figure 5.7: The AUC for GBDT, GBDT without historical interactions, and logistic regression.

marginal effect of phone calls based on PRP (see Equation (5.3) and the discussion below it). Moreover, alternatives like the logarithmic loss and accuracy depend on the distribution of the two classes (paying and non-paying debtors), which is undesirable as it varies over time.

We also compute the AUC over time by computing the AUC of each model $f_2, \ldots, f_{50}$ separately. More precisely, for each $k \in \{2, \ldots, 50\}$, we use f_k to compute the PRP at day k for each debtor in the validation set that was still in the collection process after k days since arrival. Figure 5.7 shows the AUC scores over time for the GBDT model. We also compare it against two other benchmark models: GBDT without collector-debtor interaction features (e.g., days since last phone call), and logistic regression with all features.[3] GBDT's performance improves as debtors are further in the collection process up to around day 40, and then deteriorates. In contrast, GBDT without collector-debtor interaction features has consistently lower AUC and does not improve for debtors further in the collection process. This suggests that past collector-debtor interactions have explanatory power and that the predictive power increases when more information about the debtor becomes available.

The logistic regression model exhibits a similar trend as the GBDT model in Figure 5.7. However, it consistently underperforms both GBDT models up to day 29, beyond which it surpasses the GBDT model without collector-debtor interaction features, but still underperforms the GBDT model with the complete feature set. Even though the logistic

[3] All categorical features are dummy encoded to ensure consistency across the three models.



(a) (b)

Figure 5.8: PRP over time for a debtor that did not repay (a) and that did repay (b).

regression model uses the exact same features as GBDT, it does not perform as well because it imposes a specific parametric specification on the features and cannot use information from feature interactions (e.g., the combined effect of five prior interactions and one prior phone call and eight days since the previous phone call). The AUC scores of all three models deteriorate after day 40. This is likely due to a decreasing number of training observations—by day 40 only 2,141 debtors (4.2%) remain in the training set.

5.6.3 Illustration of individual PRP trajectories

Our framework allows us to look back at individual debtor histories and observe the PRP dynamics with respect to events during the collection process. Figure 5.8 depicts the collection process for two debtors from the validation set: one which did not repay its debt (panel (a)) and one which did repay (panel (b)).

The debtor in panel (a) starts with a PRP of around 0.80, which then gradually decreases to around 0.30. Over time, no events occur that indicate that the debtor is going to settle and, consequently, the PRP declines gradually over time. It turned out that the debtor in panel (a) had moved to a different address and could not be found, explaining the unsuccessful collector-debtor interactions as they likely were not with the actual debtor.

With a PRP of approximately 0.60 initially, the debtor in panel (b) starts with a lower PRP than the debtor from panel (a). However, the PRP does not deteriorate much over time and positive jumps occur frequently. These jumps are due to the fact that events occur that positively influence its PRP. Specifically, the debtor both answers calls and makes phone calls to the Collector itself, which are indications that the debt will likely

be recovered.

5.6.4 Feature importance for predicting PRP

A natural question to ask is which features are informative in predicting PRP. In tree-based models, the (relative) importance of each of the features is not immediately observable. Unlike linear models, tree-based methods do not produce a set of coefficients that represent the (linear) effects of the features on the (predicted) outcome. A number of methods have been developed in recent times to help interpret or explain the predictions of complex machine learning models (Ribeiro et al., 2016, Lundberg and Lee, 2017), but here we rely on a simpler approach that is already included as part of the GBDT implementation. Given that trees are built by sequentially partitioning the features that have the most predictive power (see Section 5.4.3), it can be inferred as follows what the most informative features are.

Suppose we are training model f_k for $k \in \{1, \ldots, K\}$ and we are building a regression tree at Step 2 from the algorithm description in Section 5.4.3. Suppose this tree has $T \in \mathbb{N}$ terminal nodes and denote the rectangular, non-overlapping partitioning of the feature space by $R_1, \ldots, R_T$. Then, the so-called *variance gain* at a terminal node $t \in \{1, \ldots, T\}$ from splitting feature $j \in \{1, \ldots, n_f\}$, where $n_f \in \mathbb{N}$ denotes the number of features, at $d_j \in \mathbb{R}$, is equal to (Friedman, 2001, Nielsen, 2016, p.62):

$$\frac{\left(\sum_{i:x_i \in R_t, x_{ij} \leq d_j} (\sigma(z_i) - y_i)\right)^2}{2|\{i \,:\, x_i \in R_t, x_{ij} \leq d_j\}|} + \frac{\left(\sum_{i:x_i \in R_t, x_{ij} > d_j} (\sigma(z_i) - y_i)\right)^2}{2|\{i \,:\, x_i \in R_t, x_{ij} > d_j\}|} - \frac{\left(\sum_{i:x_i \in R_t} (\sigma(z_i) - y_i)\right)^2}{2|\{i \,:\, x_i \in R_t\}|}. \tag{5.5}$$

This expression approximates the gain (i.e., decrease) in the logistic loss function when, at node $t \in \{1, \ldots, T\}$, splitting feature $j \in \{1, \ldots, n_f\}$ at value $d_j \in \mathbb{R}$. The regression tree from Step 2 of Section 5.4.3 is built by iteratively splitting the feature so that (5.5) is maximized (the implementation of GBDT that we rely on (LightGBM) uses an approximation of (5.5) that is more computationally efficient).

We use the variance gain, as implemented in LightGBM (Ke et al., 2017), to analyze the importance of features by summing, for each feature, the variance gains of all the nodes in all the trees at which this feature was split. For easier interpretation, we normalize the variance gain of each feature so that the sum of all feature variance gains equals 100. Since we train one GBDT model for each of the forty-nine days (days 2 to 50) into the collection process, there are forty-nine sets of feature importances. Table 5.2 presents

Table 5.2: Feature importance

	Percent of the total variance gain of each feature					
	5 days	15 days	25 days	35 days	45 days	Average
Initial debt amount	22.02	19.16	15.62	11.82	16.30	16.24
Customer tenure	24.07	18.98	15.92	14.78	11.75	15.96
Has partially repaid debt	0.00	0.05	0.00	0.07	0.00	0.08
Repayment plan in place	0.00	0.02	0.79	0.32	0.73	0.52
Phone number is available	19.65	10.73	2.97	0.82	0.18	6.65
E-mail address is available	2.80	1.44	1.11	1.21	0.96	1.43
Product type	0.00	0.00	0.00	0.00	0.00	0.00
Amount repaid already	0.00	0.17	0.05	0.17	0.47	0.44
Collector collected from debtor before	1.80	2.74	2.03	1.03	0.99	1.71
Average income in postal code area	11.44	9.89	8.79	8.62	7.32	9.00
Share of people < 30 in postal code area	5.75	5.21	6.21	6.49	7.84	6.34
Current substatus	0.02	2.66	5.76	5.50	2.59	3.29
Passed final letter	0.00	0.00	0.00	0.40	3.22	0.90
Has answered a phone call	0.67	0.35	0.60	0.14	0.09	0.42
Promised to repay	0.03	0.90	1.55	1.13	0.22	1.12
No. of previous coll.-debtor interactions	1.47	2.34	2.38	2.65	2.67	2.32
No. of previous phone calls	0.61	1.23	1.42	1.46	1.83	1.50
Days since promise to repay	0.00	0.95	5.50	2.74	3.32	2.54
Days since last coll.-debtor interaction	0.50	1.06	2.75	8.88	9.85	4.93
Days since last phone call	0.55	1.75	2.65	3.22	4.92	2.84
Days since last answered phone call	0.64	2.12	2.76	3.37	4.84	2.73
Days since last incoming contact	0.92	2.92	5.26	7.57	6.29	4.03
Days since last incoming e-mail	0.92	1.56	2.26	2.71	1.16	1.81
Days since last incoming phone call	3.86	10.53	10.77	11.41	5.98	8.95
Day of week	1.29	2.10	1.17	1.78	1.40	1.55
Week of month	0.99	1.15	1.70	1.69	0.87	1.28

The relative percent variance gain of each feature from the state space is tabulated for GBDT models trained on debtors that are 5, 15, 25, 35, and 45 days since arrival. The average gain for each feature across all models (days since arrival of 2 to 50 days) is tabulated in the last column.

the relative variance gain of each feature for five GBDT models, namely for f_5, f_{15}, f_{25}, f_{35}, and f_{45} (corresponding to 5, 15, 25, 35, and 45 days since arrival, respectively). The average gain of each feature across all forty-nine models is also included. Note that feature importance does not specify in which direction a feature affects the model's predictions. GBDT does not assume monotonic feature effects so the same feature can have positive or negative effects under different conditions.

Two observations can be made from Table 5.2. First, the features *initial debt amount* and *customer tenure* are highly influential in predicting repayment probability of debtors, especially early in the collection process. A reason for this may be that debtors that owe more money are less likely to repay their debt, and debtors that have been customers for a longer period of time are more likely to repay their debt as they have probably had a better relationship with the debt owner. Indeed, when we compute the correlation between PRP and *initial debt amount* and *customer tenure* for debtors in the validation set for each model across all models we obtain an average of -0.26 and 0.42, respectively.

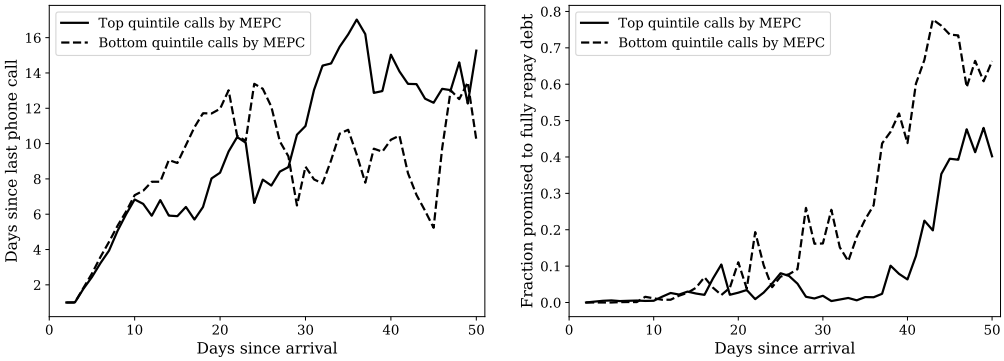
Conversely, a number of features have very low impact, such as *has partially repaid debt*, *repayment plan in place*, and *product type*. This is surprising because we expected the debtors that have already repaid part of their debt or have agreed to a repayment plan would be more likely to fully repay their debt. However, this seems to not be the case. A possibility that these features have low impact is because there is little variation in the training data. In particular, *product type* actually has zero variation in the training data as it was first recorded in January of 2016, so that all observations in the training data have the same value for this feature.

The second observation is that the impact of features varies for different days since arrival. Debtor-specific features, such as *phone number available*, tend to have relatively high impact in models that pertain to fewer days since arrival (e.g., five days). On the contrary, features related to collector-debtor interactions, such as *days since last collector-debtor interaction*, have more impact later in the collection process. The GBDT predictions also reflect this, as the correlation coefficient between *phone number available* and PRP in the validation set decreases from 0.50 at five days since arrival down to 0.03 at forty-five days since arrival, and the magnitude of correlation between *days since last collector-debtor interaction* and PRP increases from -0.02 at five days since arrival to -0.19 at forty-five days since arrival. Finally, some features, such as *current substatus* and *days since last incoming phone call*, have the bulk of their effect towards the middle of the collection process. This shows that different features predict repayment in different ways throughout the collection process.

5.6.5 Marginal effect of phone calls on PRP

Our approach to maximize the amount of debt collected is to compute the difference in PRP between making and not making an additional phone call to a debtor—see Section 5.4 and Equation (5.3) in particular. To analyze how the marginal effect of a phone call on PRP (abbreviated to MEPC) depends on the state of a debtor, we compute the MEPC for every debtor in the validation set for every day that the debtor was eligible for receiving a phone call. We identified 432,555 of such *potential* phone calls across the 28,900 debtors in the validation set over 497 days. The average MEPC over these phone calls equals 0.92%, with a standard deviation of 3.73%. This indicates that, on average, phone calls have a positive effect on PRP, but that the effect varies substantially across debtor states.

Analyzing the impact of a feature on the MEPC is non-trivial, given that features interact and correlate with each other. For example, *current substatus* and *number of previous*



(a) (b)

Figure 5.9: Average values for the top and bottom quintiles by MEPC for days since last outbound phone call (a) and fraction of debtors that promised to fully repay debt (b).

collector-debtor interactions are highly correlated as they change in similar directions with the amount of time spent in the collection process. This makes it challenging to attribute MEPC to a single feature. Instead, we present some specific insights to illustrate how MEPC can differ under different conditions. To do so, we sort the 432,555 phone calls in the validation set by MEPC and compare average feature values for each day since arrival between the top and bottom quintiles, which represent the most and least effective phone calls. Figure 5.9 contains plots for two of the features, *days since last phone call* and *promised to repay*.

For *days since last phone call* (Figure 5.9a), there is no difference between the most and least effective phone calls in the first 10 days of the collection process. From day 11 to 28, the debtors that have been called more recently seem to be better options for the next call. Starting from day 29, the effect flips around and it becomes better to call the debtors that were called less recently. Arguably, this has to do with intrinsic differences in the types of debtors that remain earlier versus later into the collection process. More debtors that are reachable and able to repay their debt remain in collections between 10 and 28 days into the process (i.e., after the first letter but before the final letter). These debtors likely require more persuasion so calling them more often could increase collectibility. As these debtors eventually repay their debt, the debtors that remain later into the collection process are more likely to either be unreachable or unable to repay their debt, and as a result, there is less value in calling them often.

For *promised to repay* (Figure 5.9b), it seems in general better to call debtors that have not previously promised to repay their debt. The fraction of debtors that have promised to repay their debt is similar between the top and bottom quintile of calls up to day 26

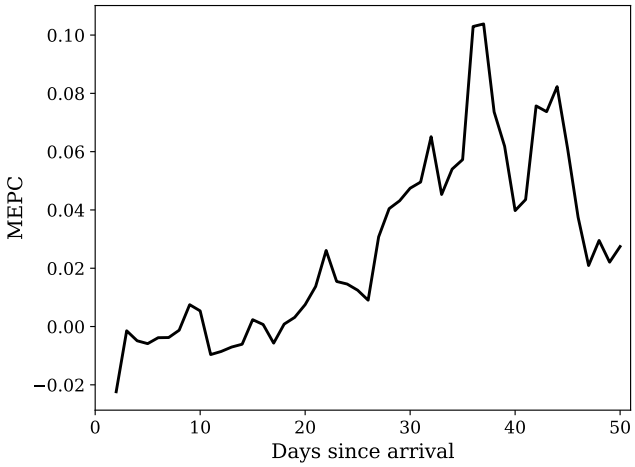


Figure 5.10: MEPC for each day since arrival.

and differs significantly afterward. The intuitive explanation for this observation is that debtors that have promised to repay their debt are already likely to repay their debt without further intervention. Until they have broken their promise, it is better to accept their promise and simply wait for repayment.

In Figure 5.10, we plot the MEPC against *days since arrival*. Phone calls seem to have limited effect until day 20, and then become increasingly effective until day 37, after which they become less effective again. Although there may be many reasons causing this, we believe the increase in effectiveness is because the debtors that are able and willing to repay their debt will likely do so early in response to the letters and e-mails, so phone calls are unnecessary and add no value to the collectibility of these cases. On the contrary, debtors that have not already repaid their debt two or three weeks into the collection process are more likely to be unwilling to repay, thus phone calls can add greater value. Phone calls begin to lose effectiveness after day 37 as more of the debtors remaining are either unreachable or are unable to repay the debt. In the former case, the debtor will be escalated to the legal phase, and in the latter case, an agreement might be reached between the Collector and a debt management intermediary. Neither of these cases will result in repayment via the collection process and spending additional phone calls on these debtors will be wasted effort.

We emphasize that both the training and validation datasets are realizations of the Collector's actual collection process and do not reflect any form of randomized experimentation. Therefore, many of the possible states may have been under-observed (e.g., zero

phone calls made within the first 15 days of the collection process). Moreover, because the collection process is path dependent and we do not know how individual collectors select which debtors to call, we do not know what would have happened had different actions been taken and cannot be sure that our MEPC estimates are unbiased. Ultimately, a controlled field experiment is necessary to understand the true value of using a data-driven prediction model to optimize phone calls.

5.7 Controlled Field Experiment

In this section, we present the results of a controlled field experiment that we conducted. First, in Section 5.7.1, we discuss how we designed the experiment. Then, in Section 5.7.2, we present the results and discuss the collection performance of our proposed policy. Finally, in Section 5.7.3, we analyze how our proposed policy differs from the incumbent policy by inferring how the (states of) debtors that our policy calls differ from those called by the incumbent policy.

5.7.1 Experimental setup

To evaluate the performance of the GBDT-optimized calling policy (GOCP), which is described in Section 5.4.2, we ran a controlled field experiment spanning a period of 102 days starting on January 19, 2018 and ending on April 30, 2018. The experiment regards debtors that are clients of the same insurance company used for the data analysis in Section 5.5 and the model development in Section 5.6.

Starting from January 19, 2018, we assigned each newly arriving debtor randomly to one of the following three collection policies: GOCP, the incumbent policy (IP) currently used by the Collector (described in Section 5.2.2), or a third experimental policy, which is not related to our research. For our analysis, we consider debtors that arrived between January 19, 2018 and February 28, 2018, since for these debtors we observe the outcome of the collection process (given that the experiment ran until April 30, 2018). As a result, a total of 921 debtors are within the scope of our experiment, of which 455 were exposed to GOCP and 466 to IP.

The implementation of GOCP is as follows. At the start of each day, we train fifty GBDT models, as described in Section 5.4.1, based on all the closed cases from January 1, 2016 up to the present day. Then, according to Equation (5.3) in Section 5.4.2, we compute the predicted marginal effect of making a phone call for each of the open cases that are

no more than fifty days into the collection process and are eligible for receiving phone calls. Debtors are not eligible for receiving phone calls if, e.g., the debt is currently being investigated because the debtor disputed the debt (we use the substatus of debtors, see Section 5.5, to verify if a debtor is eligible for receiving phone calls).

The Collector's objective is to maximize the number of cases that are completely recovered, since only in case of full recovery does the Collector receive a collection fee, which is independent of the amount of debt. To this end, when computing the marginal effect of phone calls according to Equation (5.3), we set $\text{debt}_i = 1$ for each debtor $i \in \{1, \ldots, N\}$ (note that the amount of debt is still part of the state space of a debtor, since it does affect the likelihood of repayment). In this way, GOCP is designed to maximize the number of fully recovered cases.

The cases are then sorted by the predicted marginal effect of calling, and the top 20% of these debtors is added to the pool of planned phone calls for that day. A number of phone calls following other policies (including IP) are also added to this pool. This is done in a way that the capacity of the Collector on a given day is sufficient to make all the phone calls in the calling pool. Agents in the Collector's call center are free to call debtors from the calling pool at their discretion. To prevent bias originating from behavioral changes of the agents, there is no identifier of which debtors are following which policy within the pool of planned calls and the agents were not aware of this experiment.

The Collector is required to always attempt at least one phone call before sending out the next letter. To conform with this, when a specific debtor is not called for a long time, it is artificially prioritized and scheduled for a phone call, so that the next letter can be sent afterward.

5.7.2 Experimental results

We evaluate the performance of GOCP (relative to IP) by considering the state of each of the 921 debtors in the scope of this experiment at the time the experiment ended on April 30, 2018. In Table 5.3, we provide summary statistics on the performance of IP and GOCP. In general, GOCP is able to (fully) recover more cases and collect more debt in shorter time, whilst requiring much fewer resources.

To see this, observe that IP recovered only 57.2% of the total outstanding debt, whilst GOCP was able to recover 65.2%, which is a substantial relative increase of 14.0%. This corresponded with 62.6% of the cases being fully recovered by GOCP and 59.0% by

Table 5.3: Summary statistics on the results of the experiment.

	IP	GOCP	
Key figures			
Number of debtors	466	455	
Number of outbound calls	1,119	876	
Number of inbound calls	236	188	
Fraction of total debt recovered	0.572	0.652	
Amount collected per call (€)	31.80	46.30	
Amount collected per outbound call (€)	38.53	56.72	
Descriptive statistics			
Fraction of debtors that is called at least once	0.916	0.831	***
Fraction of outbound phone calls picked up	0.268	0.223	
Fraction of fully collected cases	0.590	0.626	
Average number of days until first phone call	4.5	7.7	***
Average number of outbound calls per debtor	2.40	1.93	***
Average number of days until full repayment	22.2	20.3	
Average initial debt (€)	161.74	166.17	

***, **, and * indicate a statistically significant difference at a significance level of 0.01, 0.05, and 0.10, respectively. Inbound calls are calls made by debtors to the Collector, e.g., after missing a phone call from the Collector or after receiving one of the letters.

IP. Of the fully recovered cases, the average number of days until complete repayment is 22.2 for IP and 20.3 for GOCP, suggesting that GOCP is not only more effective at collecting more debt, but that it also does so in less time.

The total number of phone calls made by GOCP is 21.8% lower than IP (876 vs 1,119 outbound calls). Together with the increased amount of debt recovered, this leads to a 47.2% increase in the monetary amount collected per call made by GOCP as compared to IP (€56.72 vs €38.53). When also including the inbound phone calls—usually made by debtors after missing a call or receiving a letter from the Collector—we see the same picture, namely an increase in 45.6% in euros collected per call. Overall, this shows that the calls under our data-driven policy are considerably more effective in terms of return on effort than the incumbent policy.

The (decrease in) number of outbound phone calls under GOCP is mostly the consequence of the implementation of GOCP, in which the 20% of debtors is called that have the highest predicted marginal increase in PRP (see Section 5.7.1). Hence, the decrease in number of phone calls on itself is not a direct consequence of GOCP. However, it is due to GOCP that the most effective phone calls can be selected, which, in this case, leads to more debt being recovered, whilst requiring fewer phone calls.

Apart from the number of phone calls, the fraction of debtors that receive at least one phone call is also substantially lower for GOCP than for IP, namely 83.1% vs 91.6%. Presumably, this is due to the fact that, on average, GOCP makes the first phone call to a debtor only after 7.7 days, whilst IP does so after 4.5 days (recall that IP always schedules a phone call immediately after sending out the first letter). Consequently, IP calls debtors that would have settled on short-term notice without requiring the persuasion of one of the Collector's agents—thereby wasting expensive capacity. On the other hand, GOCP postpones the first call, thereby allowing debtors to settle and spending only effort on debtors that are not able or willing to pay.

All in all, the results reveal that the data-driven approach to debt collection that we propose substantially outperforms the existing policy that is based on business rules. As such, it shows that the historical data available contains sufficient information to make better decisions in a highly automated manner and that our framework is capable of operationalizing and monetizing this data.

5.7.3 Comparison of GOCP and IP

The previous section revealed that GOCP is much more effective as a calling policy than IP. In this section, we analyze how the debtors that were selected by GOCP differ from those selected by IP. To this end, we identified all phone calls that were the first phone call to a debtor on a given day. These calls are most likely a result of the respective policies and were not in response to an incoming call or a chain of phone communication within a day. Based on these calls, we made a comparison between the state of debtors called by GOCP and by IP, of which the results are presented in Table 5.4 (see Section 5.3 for a full description of the state space). For example, GOCP calls debtors that are on average 18.6 days into the collection process, whilst for IP this is 15.6 days. Recall that *telephone number available* regards to whether the phone number was known at the time the debtor arrived, i.e., whether it was provided by the debt owner. In case it is not available, phone calls can still be made if a phone number can be retrieved through, e.g., public telephone directories. In the following sections, we discuss insights derived from Table 5.4.

GOCP spends more effort on difficult cases

The results in Table 5.4 indicate that GOCP, as compared to IP, focuses on debtors that are more difficult to collect from. To see this, first observe that, compared to IP, GOCP

Table 5.4: Comparison of the state of debtors that IP and GOCP call.

Debtor state feature	IP	GOCP	
Debtor specific			
Days since arrival	15.6	18.6	***
Initial debt (€)	167.67	173.08	
Has partially repaid debt	0.005	0.006	
Repayment plan in place	0.036	0.045	
Phone number is available	0.712	0.634	***
E-mail address is available	0.765	0.706	***
Amount already repaid (percent of total)	0.002	0.001	
Collector collected from debtor before	0.368	0.326	*
Average income in postal code area (thousands)	33.35	34.28	**
Share of people under 30 in postal code area	0.361	0.361	
Passed final letter	0.001	0.000	
Historical interaction			
Has answered call before	0.231	0.162	***
Promised to repay	0.104	0.069	**
Number of previous collector-debtor interactions	3.9	3.6	**
Number of previous phone calls	1.0	0.9	*
Days since promise to repay	19.5	22.1	
Days since last collector-debtor interaction	2.8	5.9	***
Days since last phone call	13.3	13.5	
Days since last answered phone call	17.1	17.9	
Days since last incoming contact	14.6	17.4	*
Days since last incoming e-mail	12.0	16.7	**
Days since last incoming phone call	18.0	20.8	

Based on 830 and 642 calls initiated by IP and GOCP, respectively. ***, **, and * indicate a statistically significant difference at a significance level of 0.01, 0.05, and 0.10, respectively.

calls more debtors that did not answer a phone call before. This suggests that GOCP calls debtors that are more difficult to reach and likely more difficult to collect from. Second, GOCP calls less frequently to debtors that have previously promised to settle their debt. Hence, GOCP spends relatively more effort on debtors that have not yet shown willingness to pay by acknowledging their debt to the Collector. Third, GOCP calls debtors that have been in the process for a longer period of time (18.6 days vs 15.6 days), which also suggests that GOCP focuses on the more difficult cases—given that debtors with a high willingness to pay settle early in the process. Finally, compared to IP, GOCP calls more often to debtors for which less information is available. Namely, the fraction of debtors of which the phone number and e-mail address were known is lower for GOCP. The unavailability of this information makes collection work more difficult, but GOCP suggests that it is worthwhile pursuing those debtors (e.g., by searching for their phone number elsewhere).

Timing of phone calls

IP schedules debtors immediately after they receive one of the four letters that are sent by the Collector, and are then "forgotten" until the next letter has been sent. On the other hand, GOCP calls debtors at its discretion based on the debtors' states. As a result, from Table 5.4, we clearly see that the timing of phone calls made by GOCP differs from that of IP: GOCP makes on average the first phone call to a debtor 3.2 days later than IP (see Section 5.7.2) and also calls debtors that have been in the collection process longer (18.6 days for GOCP as compared to 15.6 days for IP, see Table 5.4). In addition, the number of days since last collector-debtor interaction is much higher for GOCP (5.9 days for GOCP vs 2.8 days for IP).

These observations, together with the results from Section 5.7.2, suggest the following. After sending letters, it is worthwhile to give debtors some slack to settle their debt, before spending expensive phone calls on these debtors. Hence, by waiting longer before making the first call and waiting longer after the previous interaction with the debtor, automatically only debtors that are unable or unwilling to pay remain—these are exactly the debtors that the Collector would want to spend capacity on.

Recall that the Collector requires that a phone call is made before debtors can be escalated to the next letter, thereby setting an upper bound on the number of days without a phone call (see Section 5.7.1). Without this business constraint, we expect the results to be even more profound.

Similarities between GOCP and IP

Although GOCP and IP call debtors that differ in several aspects, they are not significantly different in other aspects. It is surprising to see that debtors called by both policies do not significantly differ in outstanding debt (€173.08 vs €167.67 for GOCP and IP respectively). The difference is almost identical to the difference in initial outstanding debt for debtors assigned to the two policies. However, both policies actually select debtors with greater debt than average (€163.93). This is likely due to the fact that debtors with smaller debt amounts tend to repay earlier, and thus more calls are eventually made to debtors with larger debt amounts.

5.8 Conclusion

This chapter considers the problem of deciding on a daily basis which debtors a debt collection agency should call, given that only a limited amount of calls can be made by its agents. This is a challenging optimization problem since, at any given time, a debtor portfolio consists of a large collection of heterogeneous debtors that are at different stages in the collection process. Our approach is to formulate an MDP and approximate it through data-driven machine learning methods—thereby circumventing dimensionality issues by relying on historical data. This approach revolves around computing, for each debtor at each state, the predicted repayment probability (PRP) and inferring the marginal increase in PRP when making an additional phone call. We find that in a holdout sample our machine learning technique achieves 0.689 AUC for the PRP estimates. Furthermore, we were able to obtain an improved scheduling policy for our industry partner. We validated this policy in a controlled field experiment conducted with real debtors. The results show that our data-driven policy substantially outperforms the current scheduling policy. Most notably, our policy leads to an increase in the amount of debt collected per outbound call from €38.53 to €56.72, leading to a 47.2% improvement in return on calling effort. The improvement comes mostly from selecting debtors that have been in the collection process longer, have not been contacted recently, and have not previously answered calls nor promised to repay their debt. In general, the proposed policy puts more emphasis on debtors that are harder to collect from and calls are scheduled later in the collection process.

Summary

Many organizations strive to become *data-driven* these days. This means that the decisions they make are based on data instead of, for example, intuition, experience, or domain knowledge. The promise is that data-driven decisions are *better* decisions. Indeed, the ever-increasing amount of available data suggests that there is business value to be gained—for example, by increasing the ability to understand customers' needs or by working in a more cost-effective manner. For many organizations, however, it remains an open question as to how all of the available data can be leveraged to deliver real value. This dissertation addresses that question in several ways.

The first three chapters of this dissertation consider data-driven price optimization. This concerns the question about what the *right* price is for the goods or services that a company sells. In this context, the right price is usually the price that maximizes the profit or the turnover that it induces. Depending on the industry considered, various factors play a role in setting the right price, such as price elasticity, competition, capacity, and consumer behavior. Arguably the most apparent example is that of the fluctuating prices for airline tickets. Although consumers often disapprove with such pricing practices, it is economically justifiable: if the number of available seats on a flight decreases, the scarcity increases, and so should the price. Other examples of data-driven pricing strategies are Uber's surge pricing and the dynamically priced assortment of retail giant Amazon.

In Chapter 2, we consider the problem of optimizing the prices of an assortment of products when the relationship between price and demand is known. That is, we *know* the demand function, but the prices that maximize the revenue are unknown. The contribution of this chapter is that we consider a new demand function in the context of price optimization. This demand function—the latent class logit model—is capable of capturing customer heterogeneity, whereas existing models assume that all customers are the same. More precisely, in our approach, we acknowledge that different *types* of customers exist with different preferences.

For this demand function, we develop an algorithm that obtains the optimal prices with arbitrary precision. The novelty of our algorithm is that the computation time grows slowly ("polynomially") when the number of products in the assortment increases. This is an appealing property, as the number of products of, for example, an online retailer, is potentially enormous. By incorporating customer heterogeneity in a scalable way, Chapter 2 contributes in bridging the gap with business practice, for which many complexities still need to be addressed.

Another complexity that makes price optimization challenging in practice is the existence of competition. This, we consider in Chapter 3 by studying a market in which multiple competitors offer similar products to a heterogeneous customer base. In addition, the sellers are uncertain about the price elasticity of their customers, meaning that the seller does not know exactly how price changes will affect the demand for their products. The idea is that sellers carefully experiment with prices to "learn" the demand function and, subsequently, set the optimal price.

This scenario, i.e., dynamic pricing and learning with competition, is a notoriously difficult problem to solve. Therefore, we organized a competition in which contestants were invited to submit pricing and learning algorithms, which we then used to simulate different market environments. In doing so, we were able to obtain evidence on how pricing algorithms should be designed. Our results indicate, amongst other things, that ignoring competition is increasingly harmful when the number of competitors increases, and that an "every day, low price" strategy is very difficult to beat.

Chapter 4 is an empirical study in which we model customer behavior by relying on the transaction data of a large department store. We model how the price and brand of a product influence the decision-making process when the customer decides which product to purchase. Our model allows us to predict on an individual level which products a customer will likely purchase. We show how these predictions can be leveraged to offer personalized promotions to customers, where both the product and the discount level are customized per individual.

The final chapter of this dissertation (Chapter 5) considers the data-driven scheduling of phone calls made by a debt collection agency. Our data-driven scheduling algorithm decides on a daily basis which debtors should be called to maximize the amount of debt recovered in the long run. These phone calls are meant to inform the debtors about the amount of debt owed, persuade them to recover the debt, or to negotiate payment plans in case the debtors are willing, but unable to pay. To identify the debtors for which phone calls are most valuable, we use a state-of-the-art machine learning method to predict how the likelihood of repayment increases when a phone call is made.

To validate our methodology, we ran a large controlled field experiment with real debtors. The results indicate that our data-driven scheduling methodology leads to substantially more repayments than the policy used in practice. Not only were we able to collect more debt, it also required less time to collect it. As a result, the debt collected per outgoing phone call increased significantly.

All in all, this dissertation contributes to both the theory and practice of data-driven optimization, and in particular on data-driven pricing. It considers important open problems, such as optimal pricing with heterogeneous customers and pricing and learning with competition. Some parts (such as Chapter 2) have a strong focus on developing new theory, thereby, paving the way for future research. Other parts (for example, Chapter 5) are applied in nature and directly applicable in business practice.

Bibliography

Abe, N., Melville, P., Pendus, C., Reddy, C. K., Jensen, D. L., Thomas, V. P., Bennett, J. J., Anderson, G. F., Cooley, B. R., Kowalczyk, M., et al. (2010). Optimizing Debt Collections Using Constrained Reinforcement Learning. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 75–84. ACM.

Aghion, P., Espinosa, M. P., and Jullien, B. (1993). Dynamic duopoly with learning through market experimentation. *Economic Theory*, 3(3):517–539.

Ainslie, A. and Rossi, P. E. (1998). Similarities in choice behavior across product categories. *Marketing Science*, 17(2):91–106.

Akcay, Y., Natarajan, H. P., and Xu, S. H. (2010). Joint dynamic pricing of multiple perishable products under consumer choice. *Management Science*, 56(8):1345–1361.

Alepuz, M. D. and Urbano, A. (1999). Duopoly experimentation: Cournot competition. *Mathematical Social Sciences*, 37(2):165–188.

Alptekinoğlu, A. and Semple, J. H. (2016). The exponomial choice model: A new alternative for assortment and price optimization. *Operations Research*, 64(1):79–93.

Andersen, M., Dahl, J., and Vandenberghe, L. (2018). CVXOPT: A Python package for convex optimization, version 1.2.0. Available on: cvxopt.org.

Anufriev, M., Kopányi, D., and Tuinstra, J. (2013). Learning cycles in Bertrand competition with differentiated commodities and competing learning rules. *Journal of Economic Dynamics and Control*, 37(12):2562–2581.

Araman, V. and Caldentey, R. (2009). Dynamic pricing for nonperishable products with demand learning. *Operations Research*, 57(5):1169–1188.

Atchade, Y. F. (2006). An adaptive version for the Metropolis adjusted Langevin algorithm with a truncated drift. *Methodology and Computing in Applied Probability*, 8(2):235–254.

Aydin, G. and Porteus, E. L. (2008). Joint inventory and pricing decisions for an assortment. *Operations Research*, 56(5):1247–1255.

Aydin, G. and Ryan, J. K. (2000). Product line selection and pricing under the multinomial logit choice model. In *Proceedings of the 2000 MSOM conference*.

Belleflamme, P. and Bloch, F. (2001). Price and quantity experimentation: a synthesis. *International Journal of Industrial Organization*, 19(10):1563–1582.

Bergemann, D. and Valimaki, J. (1996). Market Experimentation and Pricing. Cowles Foundation Discussion Paper 1122, Cowles Foundation for Research in Economics at Yale University, https://ideas.repec.org/p/cwl/cwldpp/1122.html, accessed 5 June 2018.

Bertsekas, D. P. and Tsitsiklis, J. N. (1995). Neuro-dynamic programming: an overview. In *Decision and Control, 1995., Proceedings of the 34th IEEE Conference on*, volume 1, pages 560–564. IEEE.

Bertsimas, D. and Perakis, G. (2006). Dynamic pricing: a learning approach. In *Mathematical and Computational Models for Congestion Charging*, pages 45–79. Springer, New York.

Besbes, O. and Zeevi, A. (2009). Dynamic pricing without knowing the demand function: Risk bounds and near-optimal algorithms. *Operations Research*, 57(6):1407–1420.

Bhat, C. R. (1997). An endogenous segmentation mode choice model with an application to intercity travel. *Transportation Science*, 31(1):34–48.

Bischi, G. I., Chiarella, C., and Kopel, M. (2004). The long run outcomes and global dynamics of a duopoly game with misspecified demand functions. *International Game Theory Review*, 6(3):343–379.

Bischi, G. I., Naimzada, A. K., and Sbragia, L. (2007). Oligopoly games with local monopolistic approximation. *Journal of Economic Behavior & Organization*, 62(3):371–388.

Boissier, M., Schlosser, R., Podlesny, N., Serth, S., Bornstein, M., Latt, J., Lindemann, J., Selke, J., and Uflacker, M. (2017). Data-driven repricing strategies in competitive markets: An interactive simulation platform. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*, RecSys '17, pages 355–357, New York, NY, USA. ACM.

Boyd, S. and Vandenberghe, L. (2004). *Convex optimization*. Cambridge University Press.

Breiman, L. (1996). Bagging predictors. *Machine learning*, 24(2):123–140.

Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

Breiman, L., Friedman, J., Stone, C. J., and Olshen, R. A. (1984). *Classification and regression trees*. CRC press.

Broder, J. and Rusmevichientong, P. (2012). Dynamic pricing under a general parametric choice model. *Operations Research*, 60(4):965–980.

Brousseau, V. and Kirman, A. (1992). Apparent convergence of learning processes in mis-specified games. In Dutta, B., Mookherjee, D., Parthasarathy, T., Raghavan, T. E. S., Ray, D., and Tijs, S., editors, *Game Theory and Economic Applications*, pages 303–331. Springer-Verlag, Berlin.

Cartis, C., Fowkes, J. M., and Gould, N. I. (2015). Branching and bounding improvements for global optimization algorithms with Lipschitz continuity properties. *Journal of Global Optimization*, 61(3):429–457.

Chehrazi, N., Glynn, P., and Weber, T. A. (2018). Dynamic credit-collections optimization. *Management Science*. Forthcoming.

Chehrazi, N. and Weber, T. A. (2015). Dynamic valuation of delinquent credit-card accounts. *Management Science*, 61(12):3077–3096.

Chen, T. and Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794. ACM.

Chen, Y. and Farias, V. (2013). Simple policies for dynamic pricing with imperfect forecasts. *Operations Research*, 61(3):612–624.

Cheung, W. C., Simchi-Levi, D., and Wang, H. (2017). Dynamic pricing and demand learning with limited price experimentation. *Operations Research*, 65(6):1722–1731.

Chib, S., Seetharaman, P., and Strijnev, A. (2002). Analysis of multi-category purchase incidence decisions using IRI market basket data. *Advances in Econometrics*, 16:57–92.

Chintagunta, P. K., Jain, D. C., and Vilcassim, N. J. (1991). Investigating heterogeneity in brand preferences in logit models for panel data. *Journal of Marketing Research*, 28(4):417–428.

Chung, B. D., Li, J., Yao, T., Kwon, C., and Friesz, T. L. (2012). Demand learning and dynamic pricing under competition in a state-space framework. *Engineering Management, IEEE Transactions on*, 59(2):240–249.

Cole, R. (1968). *Consumer and Commercial Credit Management (Third Edition)*. Richard D. Irwin; Nobleton: Irwin-Dorsey.

Cooper, W. L., Homem-de Mello, T., and Kleywegt, A. J. (2015). Learning and pricing with models that do not explicitly incorporate competition. *Operations Research*, 63(1):86–103.

Crook, J. N., Edelman, D. B., and Thomas, L. C. (2007). Recent developments in consumer credit risk assessment. *European Journal of Operational Research*, 183(3):1447–1465.

Cyert, R. M. and DeGroot, M. H. (1970). Bayesian analysis and duopoly theory. *Journal of Political Economy*, 78(5):1168–1184.

Dasgupta, P. and Das, R. (2000). Dynamic pricing with limited competitor information in a multi-agent economy. In Scheuermann, P. and Etzion, O., editors, *Cooperative Information Systems*, volume 1901 of *Lecture Notes in Computer Science*, pages 299–310. Springer, Berlin, Heidelberg.

De Almeida Filho, A. T., Mues, C., and Thomas, L. C. (2010). Optimizing the collections process in consumer credit. *Production and Operations Management*, 19(6):698–708.

Delahaye, T., Acuna-Agost, R., Bondoux, N., Nguyen, A.-Q., and Boudia, M. (2017). Data-driven models for itinerary preferences of air travelers and application for dynamic pricing optimization. *Journal of Revenue and Pricing Management*, 16(6):621–639.

den Boer, A. V. (2015). Dynamic pricing and learning: historical origins, current research, and new directions. *Surveys in Operations Research and Management Science*, 20(1):1–18.

den Boer, A. V. and Zwart, B. (2014). Simultaneously learning and optimizing using controlled variance pricing. *Management Science*, 60(3):770–783.

den Boer, A. V. and Zwart, B. (2015). Dynamic pricing and learning with finite inventories. *Operations Research*, 63(4):965–978.

Dietterich, T. G. (2000). Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer.

DiMicco, J. M., Maes, P., and Greenwald, A. (2003). Learning Curve: a simulation-based Approach to Dynamic Pricing. *Electronic Commerce Research*, 3(3-4):245–276.

Dimitrova, M. and Schlee, E. E. (2003). Monopoly, competition and information acquisition. *International Journal of Industrial Organization*, 21(10):1623–1642.

Dong, J., Simsek, A. S., and Topaloglu, H. (2018). Pricing Problems under the Markov Chain Choice model. *Production and Operations Management*. Forthcoming.

Dong, L., Kouvelis, P., and Tian, Z. (2009). Dynamic pricing and inventory control of substitute products. *Manufacturing & Service Operations Management*, 11(2):317–339.

Duvvuri, S. D., Ansari, A., and Gupta, S. (2007). Consumers' price sensitivities across complementary categories. *Management Science*, 53(12):1933–1945.

Erdem, T. and Winer, R. S. (1998). Econometric modeling of competition: A multi-category choice-based mapping approach. *Journal of Econometrics*, 89(1):159–175.

Farias, V. and van Roy, B. (2010). Dynamic pricing with a prior on market response. *Operations Research*, 58(1):16–29.

Fawcett, T. (2006). An introduction to ROC analysis. *Pattern recognition letters*, 27(8):861–874.

Federal Reserve Bank of New York (Q1, 2018). Quarterly report on household debt and credit. Available on: https://www.newyorkfed.org/microeconomics/hhdc.html.

Fisher, M., Gallino, S., and Li, J. (2017). Competition-based dynamic pricing in online retailing: A methodology validated with field experiments. *Management Science*, 64(6):2496–2514.

Fishman, A. and Gandal, N. (1994). Experimentation and learning with networks effects. *Economics Letters*, 44(1-2):103–108.

Fowkes, J. M., Gould, N. I., and Farmer, C. L. (2013). A branch and bound algorithm for the global optimization of Hessian Lipschitz continuous functions. *Journal of Global Optimization*, 56(4):1791–1815.

Friedman, J., Hastie, T., and Tibshirani, R. (2001). *The elements of statistical learning*, volume 1. Springer series in statistics New York, NY, USA.

Friedman, J. H. (2001). Greedy function approximation: a gradient boosting machine. *Annals of statistics*, 29(5):1189–1232.

Friesz, T. L., Kwon, C., Kim, T. I., Fan, L., and Yao, T. (2012). Competitive robust dynamic pricing in continuous time with fixed inventories. Available on: https://arxiv.org/abs/1208.4374.

Gale, D. and Nikaido, H. (1965). The Jacobian matrix and global univalence of mappings. *Mathematische Annalen*, 159(2):81–93.

Gallego, A. G. (1998). Oligopoly experimentation of learning with simulated markets. *Journal of Economic Behavior & Organization*, 35(3):333–355.

Gallego, G. and Wang, R. (2014). Multiproduct price optimization and competition under the nested logit model with product-differentiated price sensitivities. *Operations Research*, 62(2):450–461.

Gelman, A. and Rubin, D. B. (1992). Inference from iterative simulation using multiple sequences. *Statistical science*, pages 457–472.

Gelman, A., Stern, H. S., Carlin, J. B., Dunson, D. B., Vehtari, A., and Rubin, D. B. (2013). *Bayesian data analysis*. Chapman and Hall/CRC.

Greene, W. H. and Hensher, D. A. (2003). A latent class model for discrete choice analysis: contrasts with mixed logit. *Transportation Research Part B: Methodological*, 37(8):681–698.

Greenwald, A. R. and Kephart, J. O. (1999). Shopbots and pricebots. In *International Workshop on Agent-Mediated Electronic Commerce*, pages 1–23. Springer.

Guadagni, P. M. and Little, J. D. (1983). A logit model of brand choice calibrated on scanner data. *Marketing Science*, 2(3):203–238.

Hanson, W. and Martin, K. (1996). Optimizing multinomial logit profit functions. *Management Science*, 42(7):992–1003.

Harrington, J. E. (1995). Experimentation and learning in a differentiated-products duopoly. *Journal of Economic Theory*, 66(1):275–288.

Harrison, J. M., Keskin, N. B., and Zeevi, A. (2012). Bayesian dynamic pricing policies: Learning and earning under a binary prior distribution. *Management Science*, 58(3):570–586.

Hetrakul, P. and Cirillo, C. (2015). Customer heterogeneity in revenue management for railway services. *Journal of Revenue and Pricing Management*, 14(1):28–49.

Hsiao, C. (2014). *Analysis of panel data*. Cambridge University Press.

Huh, W. T. and Li, H. (2015). Technical note — pricing under the nested attraction model with a multistage choice structure. *Operations Research*, 63(4):840–850.

Isler, K. and Imhof, H. (2008). A game theoretic model for airline revenue management and competitive pricing. *Journal of Revenue and Pricing Management*, 7(4):384–396.

Iyengar, R., Ansari, A., and Gupta, S. (2003). Leveraging information across categories. *Quantitative Marketing and Economics*, 1(4):425–465.

James, M. (1978). The generalised inverse. *The Mathematical Gazette*, 62(420):109–114.

Johnson Ferreira, K., Simchi-Levi, D., and Wang, H. (2016). Online network revenue management using Thompson sampling. Working paper, https://ssrn.com/abstract=2588730, accessed 23 August 2018.

Jones, E., Oliphant, T., Peterson, P., et al. (2001). SciPy: Open source scientific tools for Python, version 1.1.0. Available on: http://www.scipy.org/.

Jumadinova, J. and Dasgupta, P. (2008). Firefly-inspired synchronization for improved dynamic pricing in online markets. In *Self-Adaptive and Self-Organizing Systems, 2008. SASO '08. Second IEEE International Conference on*, pages 403–412. IEEE.

Jumadinova, J. and Dasgupta, P. (2010). Multi-attribute regret-based dynamic pricing. In Ketter, W., La Poutré, H., Sadeh, N., Shehory, O., and Walsh, W., editors, *Agent-Mediated Electronic Commerce and Trading Agent Design and Analysis*, volume 44 of *Lecture Notes in Business Information Processing*, pages 73–87. Springer Berlin Heidelberg.

Karmarkar, N. (1984). A new polynomial-time algorithm for linear programming. In *Proceedings of the sixteenth annual ACM symposium on Theory of computing*, pages 302–311. ACM.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T.-Y. (2017). Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, pages 3149–3157.

Keller, G. and Rady, S. (2003). Price dispersion and learning in a dynamic differentiated-goods duopoly. *The RAND Journal of Economics*, 34(1):138–165.

Keskin, N. B. and Zeevi, A. (2014). Dynamic pricing with an unknown demand model: Asymptotically optimal semi-myopic policies. *Operations Research*, 62(5):1142–1167.

Kim, B.-D. and Rossi, P. E. (1994). Purchase frequency, sample selection, and price sensitivity: The heavy-user bias. *Marketing Letters*, 5(1):57–67.

Kirman, A. (1983). On mistaken beliefs and resultant equilibria. In Frydman, R. and Phelps, E. S., editors, *Individual forecasting and aggregate outcomes*, pages 147–166. Cambridge University Press, New York.

Kirman, A. P. (1975). Learning by firms about demand conditions. In Day, R. H. and Graves, T., editors, *Adaptive Economics*, pages 137–156. Academic Press, New York.

Kirman, A. P. (1995). Learning in oligopoly: Theory, simulation, and experimental evidence. In Kirman, A. P. and Salmon, M., editors, *Learning and Rationality in Economics*, pages 127–178. Basil Blackwell, Cambridge, MA.

Könönen, V. (2006). Dynamic pricing based on asymmetric multiagent reinforcement learning. *International Journal of Intelligent Systems*, 21(1):73–98.

Koppelman, F. S. and Bhat, C. (2006). A self instructing course in mode choice modeling: multinomial and nested logit models. Prepared for U.S. Department of Transportation Federal Transit Administration.

Kutschinski, E., Uthmann, T., and Polani, D. (2003). Learning competitive pricing strategies by multi-agent reinforcement learning. *Journal of Economic Dynamics and Control*, 27(11-12):2207–2218.

Kwon, C., Friesz, T. L., Mookherjee, R., Yao, T., and Feng, B. (2009). Non-cooperative competition among revenue maximizing service providers with demand learning. *European Journal of Operational Research*, 197(3):981–996.

Lessmann, S., Baesens, B., Seow, H.-V., and Thomas, L. C. (2015). Benchmarking state-of-the-art classification algorithms for credit scoring: An update of research. *European Journal of Operational Research*, 247(1):124–136.

Li, G., Rusmevichientong, P., and Topaloglu, H. (2015). The d-level nested logit model: Assortment and price optimization problems. *Operations Research*, 63(2):325–342.

Li, H. and Huh, W. T. (2011). Pricing multiple products with the multinomial logit and nested logit models: Concavity and implications. *Manufacturing & Service Operations Management*, 13(4):549–563.

Li, H. and Webster, S. (2017a). Optimal pricing of correlated product options under the paired combinatorial logit model. *Operations Research*, 65(5):1215–1230.

Li, H. and Webster, S. (2017b). Product-line pricing under discrete mixed multinomial logit demand. *Manufacturing & Service Operations Management*. Forthcoming.

Li, J., Yao, T., and Gao, H. (2010). A Revenue Maximizing Strategy Based on Bayesian Analysis of Demand Dynamics. In *Proceedings of the 2009 SIAM Conference on "Mathematics for Industry"*, pages 174–181. SIAM.

Liebman, L. H. (1972). A Markov decision model for selecting optimal credit control policies. *Management Science*, 18(10):B–519.

Littlewood, K. (2005). Special issue papers: Forecasting and control of passenger bookings. *Journal of Revenue and Pricing Management*, 4(2):111–123.

Luce, R. D. (1959). *Individual Choice Behavior: A Theoretical Analysis*. John Wiley & Sons.

Lundberg, S. M. and Lee, S.-I. (2017). A unified approach to interpreting model predictions. In *Advances in Neural Information Processing Systems*, pages 4765–4774.

Ma, Y., Seetharaman, P., and Narasimhan, C. (2012). Modeling dependencies in brand choice outcomes across complementary categories. *Journal of Retailing*, 88(1):47–62.

Maddah, B. and Bish, E. K. (2007). Joint pricing, assortment, and inventory decisions for a retailer's product line. *Naval Research Logistics*, 54(3):315–330.

Manchanda, P., Ansari, A., and Gupta, S. (1999). The "shopping basket": A model for multicategory purchase incidence decisions. *Marketing Science*, 18(2):95–114.

McFadden, D., Train, K., et al. (2000). Mixed MNL models for discrete response. *Journal of Applied Econometrics*, 15(5):447–470.

Miller, G., Weatherwax, M., Gardinier, T., Abe, N., Melville, P., Pendus, C., Jensen, D., Reddy, C. K., Thomas, V., Bennett, J., et al. (2012). Tax Collections Optimization for New York State. *Interfaces*, 42(1):74–84.

Mirman, L. J., Samuelson, L., and Urbano, A. (1993). Duopoly signal jamming. *Economic Theory*, 3(1):129–149.

Mitchner, M. and Peterson, R. P. (1957). An operations-research study of the collection of defaulted loans. *Operations Research*, 5(4):522–545.

Moore, E. (1920). On the reciprocal of the general algebraic matrix. *Bulletin of the American Mathematical Society*, 26:394–395.

Ng, A. Y., Coates, A., Diel, M., Ganapathi, V., Schulte, J., Tse, B., Berger, E., and Liang, E. (2006). Autonomous inverted helicopter flight via reinforcement learning. In *Experimental Robotics IX*, pages 363–372. Springer.

Nielsen, D. (2016). Tree Boosting With XGBoost—Why Does XGBoost Win "Every" Machine Learning Competition? Master's thesis, Norwegian University of Science and Technology.

Nocedal, J. and Wright, S. J. (1999). *Numerical Optimization*. Springer, New York, NY, USA, second edition.

Olson, R. S., La Cava, W., Mustahsan, Z., Varik, A., and Moore, J. H. (2018). Data-driven advice for applying machine learning to bioinformatics problems. In *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, volume 23, page 192. NIH Public Access.

Patrick, J., Puterman, M. L., and Queyranne, M. (2008). Dynamic multipriority patient scheduling for a diagnostic resource. *Operations Research*, 56(6):1507–1525.

Penrose, R. (1955). A generalized inverse for matrices. In *Mathematical proceedings of the Cambridge philosophical society*, volume 51, pages 406–413. Cambridge University Press.

Perakis, G. and Sood, A. (2006). Competitive multi-period pricing for perishable products: A robust optimization approach. *Mathematical Programming*, 107(1):295–335.

Pintér, J. D. (2013). *Global optimization in action: continuous and Lipschitz optimization: algorithms, implementations and applications*, volume 6. Springer Science & Business Media.

Powell, W. B. and Topaloglu, H. (2006). Approximate dynamic programming for large-scale resource allocation problems. In *Models, Methods, and Applications for Innovative Decision Making*, pages 123–147. INFORMS.

Ramezani, S., Bosman, P. A. N., and La Poutre, H. (2011). Adaptive strategies for dynamic pricing agents. In *2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology*, pages 323–328. IEEE.

Rassenti, S., Reynolds, S. S., Smith, V. L., and Szidarovszky, F. (2000). Adaptation and convergence of behavior in repeated experimental Cournot games. *Journal of Economic Behavior & Organization*, 41(2):117–146.

Rayfield, W. Z., Rusmevichientong, P., and Topaloglu, H. (2015). Approximation Methods for Pricing Problems Under the Nested Logit Model with Price Bounds. *INFORMS Journal on Computing*, 27(2):335–357.

Ribeiro, M. T., Singh, S., and Guestrin, C. (2016). Why Should I Trust You? Explaining the Predictions of Any Classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 1135–1144. ACM.

Roberts, G. O., Gelman, A., Gilks, W. R., et al. (1997). Weak convergence and optimal scaling of random walk Metropolis algorithms. *The Annals of Applied Probability*, 7(1):110–120.

Roberts, G. O., Tweedie, R. L., et al. (1996). Exponential convergence of Langevin distributions and their discrete approximations. *Bernoulli*, 2(4):341–363.

Rosenthal, J. S. et al. (2011). Optimal proposal distributions and adaptive MCMC. *Handbook of Markov Chain Monte Carlo*, pages 93–112.

Rossi, P. E., Allenby, G. M., and McCulloch, R. (2012). *Bayesian Statistics and Marketing*. John Wiley & Sons.

Russell, G. J. and Kamakura, W. A. (1998). Modeling multiple category brand preference with household basket data. *Journal of Retailing*, 73(4):439–461.

Scarpa, R. and Thiene, M. (2005). Destination choice models for rock climbing in the Northeastern Alps: a latent-class approach based on intensity of preferences. *Land Economics*, 81(3):426–444.

Schinkel, M. P., Tuinstra, J., and Vermeulen, D. (2002). Convergence of Bayesian learning to general equilibrium in mis-specified models. *Journal of Mathematical Economics*, 38(4):483–508.

Seetharaman, P., Chib, S., Ainslie, A., Boatwright, P., Chan, T., Gupta, S., Mehta, N., Rao, V., and Strijnev, A. (2005). Models of multi-category choice behavior. *Marketing Letters*, 16(3-4):239–254.

Shen, J. (2009). Latent class model or mixed logit model? A comparison by transport mode choice data. *Applied Economics*, 41(22):2915–2924.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, 529(7587):484–489.

Singh, V. P., Hansen, K. T., and Gupta, S. (2005). Modeling preferences for common attributes in multicategory brand choice. *Journal of Marketing Research*, 42(2):195–209.

Song, J.-S. and Xue, Z. (2007). Demand management and inventory control for substitutable products. Working paper, Duke University, Durham.

SURFsara (2015). Lisa system. https://userinfo.surfsara.nl/systems/lisa.

Sutton, R. S. and Barto, A. G. (1998). *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge.

Tesauro, G. and Kephart, J. O. (2002). Pricing in agent economies using multi-agent Q-learning. *Autonomous Agents and Multi-Agent Systems*, 5(1):289–304.

Train, K. E. (2009). *Discrete choice methods with simulation*. Cambridge University Press.

Tuinstra, J. (2004). A price adjustment process in a model of monopolistic competition. *International Game Theory Review*, 6(3):417–442.

van de Geer, R. and Bhulai, S. (2018). Modeling similarities in choice behavior across product categories at an apparel retailer. Submitted.

van de Geer, R. and den Boer, A. V. (2018). Price Optimization Under the Finite-Mixture Logit Model. Submitted. Available at https://ssrn.com/abstract=3235432.

van de Geer, R., den Boer, A. V., Bayliss, C., Currie, C., Ellina, A., Esders, M., Haensel, A., Lei, X., Maclean, K. D., Martinez-Sykora, A., Riseth, A. N., Ødegaard, F., and Zachariades, S. (2018a). Dynamic pricing and learning with competition: insights from the dynamic pricing challenge at the 2017 INFORMS RM & pricing conference. *Journal of Revenue and Pricing Management*. Forthcoming.

van de Geer, R., Wang, Q., and Bhulai, S. (2018b). Data-Driven Consumer Debt Collection via Machine Learning and Approximate Dynamic Programming. Submitted. Available at https://ssrn.com/abstract=3250755.

Wang, R. (2012). Capacitated assortment and price optimization under the multinomial logit model. *Operations Research Letters*, 40(6):492–497.

Wen, C.-H. and Lai, S.-C. (2010). Latent class models of international air carrier choice. *Transportation Research Part E: Logistics and Transportation Review*, 46(2):211–221.

Zhang, H., Rusmevichientong, P., and Topaloglu, H. (2018). Technical note — multiproduct pricing under the generalized extreme value models with homogeneous price sensitivity parameters. *Operations Research*. Forthcoming.