Data, Models and Decisions for Large-Scale Stochastic Optimization Problems

by

Velibor V. Mišić

B.A.Sc., University of Toronto (2010)
M.A.Sc., University of Toronto (2012)

Submitted to the Sloan School of Management
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Operations Research

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2016

Author .....
Sloan School of Management
May 10, 2016

Certified by .....
Dimitris Bertsimas
Boeing Professor of Operations Research
Co-Director, Operations Research Center
Thesis Supervisor

Accepted by .....
Patrick Jaillet
Dugald C. Jackson Professor of Electrical Engineering
and Computer Science
Co-Director, Operations Research Center

Data, Models and Decisions for Large-Scale Stochastic Optimization Problems

by

Velibor V. Mišić

Submitted to the Sloan School of Management
on May 10, 2016, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Operations Research

Abstract

Modern business decisions exceed human decision making ability: often, they are of a large scale, their outcomes are uncertain, and they are made in multiple stages. At the same time, firms have increasing access to data and models. Faced with such complex decisions and increasing access to data and models, how do we transform data and models into effective decisions? In this thesis, we address this question in the context of four important problems: the dynamic control of large-scale stochastic systems, the design of product lines under uncertainty, the selection of an assortment from historical transaction data and the design of a personalized assortment policy from data.

In the first chapter, we propose a new solution method for a general class of Markov decision processes (MDPs) called decomposable MDPs. We propose a novel linear optimization formulation that exploits the decomposable nature of the problem data to obtain a heuristic for the true problem. We show that the formulation is theoretically stronger than alternative proposals and provide numerical evidence for its strength in multiarmed bandit problems.

In the second chapter, we consider to how to make strategic product line decisions under uncertainty in the underlying choice model. We propose a method based on robust optimization for addressing both parameter uncertainty and structural uncertainty. We show using a real conjoint data set the benefits of our approach over the traditional approach that assumes both the model structure and the model parameters are known precisely.

In the third chapter, we propose a new two-step method for transforming limited customer transaction data into effective assortment decisions. The approach involves estimating a ranking-based choice model by solving a large-scale linear optimization problem, and solving a mixed-integer optimization problem to obtain a decision. Using synthetic data, we show that the approach is scalable, leads to accurate predictions and effective decisions that outperform alternative parametric and non-parametric approaches.

In the last chapter, we consider how to leverage auxiliary customer data to make

personalized assortment decisions. We develop a simple method based on recursive partitioning that segments customers using their attributes and show that it improves on a "uniform" approach that ignores auxiliary customer information.

Thesis Supervisor: Dimitris Bertsimas
Title: Boeing Professor of Operations Research
Co-Director, Operations Research Center

Acknowledgments

First, I want to thank my advisor, Dimitris Bertsimas, for his outstanding guidance over the last four years. Dimitris: it has been a great joy to learn from you and to experience your unbounded energy, love of research and positivity, which continue to amaze me as much today as when we had our first research meeting. I am extremely grateful and indebted to you for your commitment to my academic and personal development and for all of the opportunities you have created for me. Most of all, your belief in the power of research to have impact has always been inspiring to me, and I shall carry it with me as I embark on the next chapter of my academic career.

I would also like to thank my committee members, Georgia Perakis and Retsef Levi, for providing critical feedback on this research and for their support on the academic job market. Georgia: thank you for your words of support, especially at a crucial moment early in my final year, and for all of your help, especially with regard to involving me in 15.099, providing multiple rounds of feedback on my job talk and helping me prepare for pre-interviews at INFORMS. Retsef: thank you for pushing me to be better, to think more critically of my work and for suggesting interesting connections to other research.

I have also many other faculty and staff at the ORC to thank. Thank you to the other faculty with whom I interacted and learned from during my PhD: thank you Patrick Jaillet, Juan Pablo Vielma, David Gamarnik, Jim Orlin, Tauhid Zaman and Karen Zheng. In addition, I must thank the ORC administrative staff, Laura Rose and Andrew Carvalho, for making the ORC a paragon of efficiency and for always having a solution to every administrative problem or question that I brought to them.

I would also like to acknowledge and thank my collaborators from MIT Lincoln Laboratories for their support: Dan Griffith and Mykel Kochenderfer in the first two years of my PhD, and Allison Chang in the last two years.

The ORC community has been a wonderful home for the last four years, and I was extremely lucky to make many great friends during my time here. In particular, thank you to Anna Papush, Alex Remorov, Alex Sahyoun, Andrew Li, Charles

Thraves, Stefano Tracà, Alex Weinstein, Miles Lubin, Chiwei Yan, Matthieu Monsch, Allison O'Hair, André Calmon, Gonzalo Romero, Florin Ciocan, Adam Elmachtoub, Ross Anderson, Vishal Gupta, Phebe Vayanos, Angie King, Fernanda Bravo, Kris Johnson, Maxime Cohen, Nathan Kallus, Will Ma, Nataly Youssef, David Fagnan, John Silberholz, Paul Grigas, Iain Dunning, Leon Valdes, Martin Copenhaver, Daniel Chen, Arthur Flajolet, Zach Owen, Joey Huchette, Nishanth Mundru, Ilias Zadik, Colin Pawlowski and Rim Hariss. Andrew and Charlie: thank you for being there from even before the beginning and for many epic nights. Anna, Alex R. and Charlie: thank you for being the best project partners ever. Alex S., Alex W. and Anna: I will never forget all the fun we had planning events as INFORMS officers. Vishal: thank you for your mentorship during my first two years and for many insightful conversations on research and life in general. Maxime and Paul: thank you for an unforgettable trip to upstate New York (and please remember to not keep things "complicated")! Vishal, Maxime and Kris: thank you for your support during the job market, especially in the last stage of the process. Nishanth: thank you for many creative meetings for the TRANSCOM project that often digressed into other research discussions. Nataly: thank you for your encouragement in difficult times and also for having plenty of embarrassing stories to share about me at social gatherings. Adam and Ross: thank you for hosting me so many years ago and setting the right tone for my ORC experience.

I also had many great teaching experiences at MIT, for which I have many people to thank. Allison: thank you for being a fantastic instructor and coordinator, and for your saint-like patience when I was a teaching assistant for the Analytics Edge in its regular MBA, executive MBA and MOOC incarnations. Iain, John, Nataly and Angie: thank you for being great teammates on the 15.071x MOOC.

Thank you to the Natural Sciences and Engineering Research Council (NSERC) of Canada for providing financial support in the form of a PGS-D award for the first three years of my PhD. Thank you also to Olivier Toubia, Duncan Simester and John Hauser for making the data from their paper [91] available, which was used in Chapter 3.

I had the great fortune of making an amazing group of Serbian friends in Cambridge during my time at MIT. To Saša Misailović, Danilo Mandić [1], Marija Mirović, Ivan Kuraj, Marija Simidjievska, Irena Stojkov, Enrico Cantoni, Miloš Cvetković, Aca Milićević, Marko Stojković and Vanja Lazarević-Stojković: thank you for everything. I will miss the brunches at Andala and the nights of rakija, palačinke, kiflice, the immortal "Srbenda" meme, the incomprehensible arguments between Marija S. and Kafka, and the MOST-sponsored Toscanini-induced ice cream comas.

I am also indebted to many friends outside of the ORC at MIT; a special thank you to Felipe Rodríguez for being a wonderful roommate during the middle of my PhD and to Francesco Lin for being a great party and concert buddy. Thank you Peter Zhang and Kimia Ghobadi for many great get-togethers to relive our old glory days at MIE (which reminds me: we still need to catch up!). Outside of MIT, thank you Auyon Siddiq for being a great friend and for always helping me in my many dilemmas, and thank you Birce Tezel for your words of reassurance in hard times and for helping me bounce gift ideas off you. I also need to thank my friends from my undergrad days in Toronto for being my biggest fans: thank you Eric Bradshaw, Jordan DiCarlo, Geoffrey Vishloff, Lily Qiu, Torin Gillen, Emma Tsui, Oren Kraus, Archis Deshpande and Anne Zhang. Thank you also to Konstantin Shestopaloff for many great lunches and inspiring conversations during my visits back to Toronto.

I would not have reached this point without the support of my family: my father Vojislav, my mother Jelena, my brother Bratislav and my sister-in-law Elena. Thank you for your unwavering and unconditional love; words cannot express how grateful I am to you for supporting me in my decision to come to MIT, for advising me in complex and delicate situations, for always being on my team, for making me feel better after particularly rough days and for visiting me in Cambridge many times over the last four years. The best work in this thesis happened during my visits to Toronto, when we would all be working on our own thing in the living room but together, and I always look forward to our next discussion, whether it is about what happened in the latest episode of *Game of Thrones* or *Državni Posao*, or Meda and

[1] Also known as "Kafka".

Mačiko's latest exploits. Which reminds me: I should also thank our quadrupedal companions, Mačiko[2] and Meda[3], for their contributions to this thesis.

Last but not least, I would like to thank my girlfriend Dijana, for her love and support, for being by my side at the lowest and the highest points of my PhD, for always being positive and for reminding me that there is more to life than research, such as getting brunch on a Sunday at Tatte or going for a run along the Charles River. Above all, thank you for making the last couple of years the happiest of my life. It is to her and my family that I dedicate this thesis.

[2]Also known as "Šmica", "Dlakavo đubre"; a black cat of unknown provenance.

[3]Also known as "Kerenko", "Keroslav", "Kerenski", "Skot"; a black lab-husky puppy. My walks with him in -15° C Toronto weather led to some of the ideas in Chapter 4.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Modern business decisions exhibit an unprecedented level of complexity. Consider the following examples:

- Given a set of projects evolving stochastically, how should we dynamically allocate our firm's resources to these projects to maximize the long term benefit to the firm?
- What variations of a new product should we offer to a customer population?
- How should we modify our product offerings to maximize revenues given aggregated transaction data from a collection of customers?
- In an online retail setting, how do we leverage information about our customers to tailor our product offerings to each individual customer?

These decisions, and many other decisions like these ones, are extremely complex. Often, the system that provides the context for our decision is one that is high-dimensional; that exhibits known, stochastic dynamics or otherwise, unknown dynamics that lead to uncertainty in our decision; and ones that evolve over multiple stages in time, requiring us to make not a single, one-shot decision, but a sequence of decisions. Such decisions exceed human decision making ability; managerial intuition and insight cannot be relied on as the sole basis for such decisions.

At the same time, businesses have increasing access to data and models that allow them to understand the effects of different decisions. For example, in the case of product line decisions, firms increasingly rely on a technique known as conjoint analysis to survey customers, which involves asking representative customers to choose from hypothetical products. Using this data on their choices, the firm can then build models to predict how customers will choose from hypothetical product offerings.

However, despite this increasing access to data and models, firms lack the capability to bridge the chasm between data/models and effective decisions. This is what the burgeoning field of *analytics* is concerned: how to transform data and predictive models into decisions that create value.

This thesis is concerned with developing new analytics methods for making complex decisions in the presence of either data or models that describe the system. These methods are largely based on modern optimization, more specifically, linear, mixed-integer and robust optimization. In what follows, we provide a high-level overview of each chapter of the thesis. A more extensive discussion of the contributions and the relevant literature can be found at the beginning of each chapter.

1.1 Decomposable Markov decision processes: a fluid optimization approach

In Chapter 2, we consider the problem of solving decomposable Markov decision processes (MDPs). Decomposable MDPs are problems where the stochastic system can be decomposed into multiple individual components. Although such MDPs arise naturally in many practical applications, they are often difficult to solve exactly due to the enormous size of the state space of the complete system, which grows exponentially with the number of components.

In this chapter, we propose an approximate solution approach to decomposable MDPs that is based on re-solving a fluid linear optimization formulation of the problem at each decision epoch. This formulation tractably approximates the problem by

modeling transition behavior at the level of the individual components rather than the complete system. We prove that our fluid formulation provides a tighter bound on the optimal value function than three state-of-the-art formulations: the approximate linear optimization formulation, the classical Lagrangian relaxation formulation and a novel, alternate Lagrangian relaxation that is based on relaxing an action consistency constraint. We provide a numerical demonstration of the effectiveness of the approach in the area of multiarmed bandit problems, where we show that our approach provides near optimal performance and outperforms state-of-the-art algorithms.

1.2 Robust product line design

In Chapter 3, we consider the problem of designing product lines that are immunized to uncertainty in customer choice behavior. The majority of approaches to product line design that have been proposed by marketing scientists assume that the underlying choice model that describes how the customer population will respond to a new product line is known precisely. In reality, however, marketers do not precisely know how the customer population will respond and can only obtain an estimate of the choice model from limited conjoint data.

In this chapter, we propose a new type of optimization approach for product line design under uncertainty. Our approach is based on the paradigm of robust optimization where, rather than optimizing the expected revenue with respect to a single model, one optimizes the worst-case expected revenue with respect to an uncertainty set of models. This framework allows us to account for both parameter uncertainty, when we may be confident about the type of model structure but not about the values of the parameters, and structural uncertainty, when we may not even be confident about the right model structure to use to describe the customer population.

Through computational experiments with a real conjoint data set, we demonstrate the benefits of our approach in addressing parameter and structural uncertainty. With regard to parameter uncertainty, we show that product lines designed without ac-

counting for parameter uncertainty are fragile and can experience worst-case revenue losses as high as 23%, and that the robust product line can significantly outperform the nominal product line in the worst-case, with relative improvements of up to 14%. With regard to structural uncertainty, we similarly show that product lines that are designed for a single model structure can be highly suboptimal under other structures (worst-case losses of up to 37%), while a product line that optimizes against the worst of a set of structurally distinct models can outperform single-model product lines by as much as 55% in the worst-case and can guarantee good aggregate performance over structurally distinct models.

1.3 Data-driven assortment optimization

In Chapter 4, we consider the problem of assortment optimization using historical transaction data from previous assortments. Assortment optimization refers to the problem of selecting a set of products to offer to a group of customers so as to maximize the revenue that is realized when customers make purchases according to their preferences. Assortment optimization is essential to a wide variety of application domains that includes retail, online advertising and social security; however, it is challenging in practice because one typically has limited data on customer choices on which to base the decision.

In this chapter, we present a two-step approach to making effective assortment decisions from transaction data. In the first step, we use the data to estimate a generic ranking-based model of choice that is able to represent any choice model based on random utility maximization. In the second step, using the estimated model, we find the optimal assortment by solving a mixed-integer optimization (MIO) problem that is scalable and that is flexible, in that it can easily accommodate constraints.

We show through computational experiments with synthetic data that (1) our MIO model is practically tractable and can be solved to full optimality for large numbers of products in operationally feasible times; (2) our MIO model is able to accommodate realistic constraints with little impact to solution time; (3) our esti-

mation procedure is computationally efficient and produces accurate out-of-sample predictions of the true choice probabilities; and (4) by combining our estimation and optimization procedures, we are able to find assortments that achieve near-optimal revenues that outperform alternative parametric and non-parametric approaches.

1.4 Personalized assortment planning via recursive partitioning

In Chapter 5, we consider the problem of making personalized assortment decisions using data. Previously, in Chapter 4, we considered the problem of how to make an assortment decision using historical data that only focuses on previous assortments and previous purchases, and does not assume that there is data about the customers who make those purchase decisions. In an increasing number of real-world assortment contexts, we have access to information about a customer. For example, in an online retail setting, we know information about the user browsing through the product offerings, such as their location (through their ZIP code for example), age, gender and many other attributes. Given this information, how can we leverage this information to make a more effective assortment decision for each individual customer?

In this chapter, we present an approach for using auxiliary customer data to make better assortment decisions. The approach is based on building a predictive model using recursive partitioning: the customers represented in the transactions are iteratively divided up according to their attributes, and the end result is a tree representation of the customers, where each leaf corresponds to a collection of customers that share similar attributes and whose choice behavior is sufficiently homogeneous. In a way, this tree can be thought of as a segmentation of the customers.

Using synthetic data, we compare our partitioning-based approach to a uniform approach, which offers the same assortment to all customers and ignores the customer attribute data, and to a "perfect foresight" approach, which knows the ground truth model perfectly and optimizes exactly for that model. We show that our ap-

proach leads to out-of-sample revenues that improve on the uniform approach, and are moderately close to revenues obtained with perfect foresight.

Chapter 2

Decomposable Markov decision processes: a fluid optimization approach

2.1 Introduction

Many real world problems involving the control of a stochastic system can be modeled as Markov decision processes (MDPs). In a typical MDP, the system begins in a certain state $\mathbf{s}$ in some state space $\mathcal{S}$. The decision maker selects an action a from some action space $\mathcal{A}$. The system then transitions randomly to a new state $\mathbf{s}'$ with probability $p_a(\mathbf{s}, \mathbf{s}')$ and the decision maker garners some reward $g_a(\mathbf{s})$. Once the system is in this new state $\mathbf{s}'$, the decision maker once again selects a new action, leading to additional reward and causing the system to transition again. In the most basic form of the problem, the decision maker needs to make decisions over an infinite horizon, and the rewards accrued over this infinite horizon are discounted in time according to a discount factor $\beta \in (0, 1)$. The goal of the decision maker, then, is to find a policy π that prescribes an action $\pi(\mathbf{s})$ for each state $\mathbf{s}$ so as to maximize

the expected total discounted reward

$$\mathbb{E}\left[\sum_{t=1}^{\infty} \beta^{t-1} g_{\pi(\mathbf{s}(t))}(\mathbf{s}(t))\right],$$

where $\mathbf{s}(t)$ is the random variable representing the state at time t, when the system is operated according to policy π. In other types of problems, the decision maker may only be making decisions over a finite time horizon; in those problems, the policy prescribing the action to take may not only depend on the state, but also on the time at which the decision is being made.

Although problems that are represented in this form can in principle be solved exactly with dynamic programming, this is often practically impossible. Exact methods based on dynamic programming require one to compute the optimal value function J^*, which maps states in the state space $\mathcal{S}$ to the optimal expected discounted reward when the system starts in that state. For many problems of practical interest, the state space $\mathcal{S}$ is so large that operating on, or even storing the value function J^*, becomes computationally infeasible. This is what is often referred to in the dynamic programming and MDP literature as the *curse of dimensionality* [8].

Where does the curse of dimensionality come from? That is, why is it that practical MDPs often have prohibitively large state spaces? For many practical problems, the system that is being modeled is often not a single, atomic system, but rather consists of a collection of smaller sub-systems or components. Mathematically, consider a system consisting of M components, where each component $m \in \{1, \ldots, M\}$ is endowed with a state s^m from an ambient state space $\mathcal{S}^m$. To represent the *complete* system, we must represent the state $\mathbf{s}$ as an M-tuple of the component states, i.e., $\mathbf{s} = (s^1, \ldots, s^M)$, and as a result, the state space of the complete system becomes the Cartesian product of the component state spaces, i.e., $\mathcal{S} = \mathcal{S}^1 \times \cdots \times \mathcal{S}^M$. As the number of components M grows, the size of the state space of the complete system grows in an exponential fashion.

At the same time, the data of such systems are often not presented to us in terms of the complete system state. The probabilistic dynamics induced by each

candidate action in $\mathcal{A}$ may be naturally expressed in terms of individual components or small combinations (e.g., pairs) of components. Similarly, the reward structure of the problem does not need to be specified in terms of the complete system state, but can be specified in terms of the component states. In the remainder of the chapter, we will refer to MDPs where the probabilistic dynamics and reward structure can be expressed in terms of the component states as *decomposable* MDPs.

Many practically relevant MDPs can be modeled as decomposable MDPs. One major class of MDPs that falls into the decomposable MDP framework is the class of multiarmed bandit problems. In the multiarmed bandit problem, the decision maker is presented with M machines ("bandits"), where each bandit m is initially in some state s^m from its state space $\mathcal{S}^m$. At each point in time, one of the bandits may be activated, in which case the chosen bandit changes state probabilistically and the decision maker earns some reward. The problem is then to decide, at each point in time, given the state of all of the bandits, which bandit to activate, so as to maximize the total expected long term reward. In the basic form of the problem – the *regular* multiarmed bandit problem – when a bandit m is activated, the inactive bandits do not change state. In the *restless* bandit problem, the inactive bandits can also change state passively and the decision maker may earn a passive reward from bandits that are not activated. The multiarmed bandit, by its definition, is a decomposable MDP: the state space of the ensemble of bandits is the product of the state spaces of the individual bandits, and the probability transition structure is specified at the level of each bandit.

In this chapter, we propose a new fluid optimization approach for approximately solving decomposable MDPs. The centerpiece of our approach is a linear optimization (LO) model in which, in its most basic form, the decision variables represent the marginal probabilities of each individual component being in each of its possible states and the action taken at a particular time, and the main constraints are conservation constraints that govern how these marginal probabilities "flow" from component states at time t to new states at $t + 1$ under different action (hence the name *fluid*). The idea of the formulation is to approximate the behavior of the system when it is

controlled optimally. The formulation achieves this in a tractable way by exploiting the decomposable nature of the problem: rather than modeling the precise transition behavior of the system at the level of tuples of component states, it models the macroscopic transitions of the system at the level of the individual components and their states. The optimal solution of the formulation, when it includes constraints that model the complete system starting in a certain state, can be used to derive an action for the state. In this way, the formulation leads naturally to a simple heuristic for solving the MDP.

Our contributions are as follows:

1. We propose a novel LO formulation for approximately modeling decomposable MDPs and an associated heuristic for solving the MDP. The formulation is tractable since the number of variables scales linearly with the number of individual components, as opposed to the exponential scaling that is characteristic of dynamic programming. We show that this formulation provides an upper bound on the optimal value of the MDP and provide idealized conditions under which our fluid formulation-based heuristic is optimal. We discuss how this basic, "first-order" formulation that models individual components can be extended to "higher-order" formulations that model combinations of components (e.g., a second-order formulation that models transitions of pairs of components). We also discuss how the basic formulation can be extended to address finite horizon, time-dependent problems.

2. We theoretically compare our fluid formulation to three alternative proposals. In particular, we show that a finite version of our formulation that models the evolution of the system over a horizon of T periods provides provably tighter bounds on the optimal value function than three state-of-the-art formulations: the approximate linear optimization (ALO) formulation of [38], the classical Lagrangian relaxation (CLR) formulation of [2] and an alternate Lagrangian relaxation (ALR) that involves relaxing an action consistency constraint. The latter alternate Lagrangian relaxation is a novel formulation that is equivalent

to the ALO and is of independent interest. Moreover, the fluid bound is non-increasing with the time horizon T. Letting $J^*(\mathbf{s})$ denote the optimal value function at the state $\mathbf{s}$, $Z_T^*(\mathbf{s})$ denote the objective value of the fluid formulation with horizon T at $\mathbf{s}$, and $Z_{ALO}^*(\mathbf{s})$, $Z_{ALR}^*(\mathbf{s})$,and $Z_{CLR}^*(\mathbf{s})$ denote the objective values of the ALO, ALR and CLR formulations at $\mathbf{s}$, respectively, our results can be summarized in the following statement, which holds for any $T \in \{1, 2, \dots\}$:

$$J^*(\mathbf{s}) \leq Z_T^*(\mathbf{s}) \leq \dots \leq Z_2^*(\mathbf{s}) \leq Z_1^*(\mathbf{s}) \leq Z_{ALO}^*(\mathbf{s}) = Z_{ALR}^*(\mathbf{s}) \leq Z_{CLR}^*(\mathbf{s}).$$

In this way, our work contributes to the overall understanding of the fluid approach and all previous proposals in a unified framework.

3. We demonstrate the effectiveness of our approach computationally on multi-armed bandit problems. We consider regular bandit problems (inactive bandits do not change state) and restless bandit problems (inactive bandits may change state). We show that bounds from our approach can be substantially tighter than those from state-of-the-art approaches, and that the performance of our fluid policy is near-optimal and outperforms policies from state-of-the-art approaches.

The rest of this chapter is organized as follows. In Section 2.2, we review the extant body of research related to this chapter. In Section 2.3, we define the decomposable MDP and present our infinite LO fluid formulation. We prove a number of properties of this formulation, and motivated by the properties of this infinite LO formulation, we present a heuristic policy for generating actions at each decision epoch based on a finite version of this formulation. In Section 2.4, we compare the finite fluid formulation to the ALO formulation of [38], the classical Lagrangian relaxation approach of [2] and the alternate Lagrangian relaxation. In Section 2.5, we apply our framework to the multiarmed bandit problem and provide computational evidence for the strength of our approach in this class of problems. Finally, in Section 2.6, we state the conclusions of our study and offer some directions for future work.

2.2 Literature review

Markov decision processes have a long history, tracing back to the work of [7] in the 1950s. The importance and significance of this area in the field of operations research is underscored both by the number of research papers that have been written in this area, as well as the numerous books written on the subject [60, 59, 75, 12].

While MDPs can be solved exactly through methods such as value iteration, policy iteration and the LO approach (see, e.g., [75]), these approaches become intractable in high-dimensional problems. As a result, much research has been conducted in the area of *approximate* dynamic programming (ADP). The interested reader is referred to [94] for a brief overview, and to [13] and [74] for more comprehensive treatments of the topic. The goal of ADP is to find an approximation to the true value function. By then applying the policy that is greedy with respect to this approximate value function, one hopes to achieve performance that is close to that of the true optimal policy.

Within the ADP literature, our work is most closely related to the approximate linear optimization (ALO) approach of [38] and the Lagrangian relaxation approach of [2]. In the ALO approach to ADP, one approximates the value function as the weighted sum of a collection of basis functions, and solves the LO formulation of the MDP with this approximate value function in place of the true value function. By doing so, the number of variables in the problem is significantly reduced, leading to a more tractable problem. In many applications, one can exploit the decomposable nature of the problem in selecting a basis function architecture: for example, in [38] the approach is applied to a queueing control example, where the value function is approximated as a linear combination of all polynomials up to degree 2 of the individual queue lengths of the system. On the other hand, [58] and [2] study MDPs where the problem can be viewed as a collection of subproblems, and the action that can be taken in each subproblem is constrained by a global linking constraint that couples the subproblems together. By dualizing the linking constraint, the complete problem decomposes along the subproblems, leading to an optimization problem that

is significantly simpler than the exact LO model of the MDP. By solving this optimization problem, one obtains an upper bound on the optimal value at a given state and a value function approximation.

Our fluid approach is closely related to the Lagrangian relaxation approach and builds on it in two important ways. First, we delineate two different types of Lagrangian relaxations: the "classical" Lagrangian relaxation, where the action space is implicitly defined by a coupling constraint, and a novel, "alternate" Lagrangian relaxation where the components are coupled by an "action consistency" constraint (the action taken in each component must be the same). Our formulation is not related to the former classical formulation, but to the latter alternate formulation. This alternate Lagrangian relaxation is significant because it extends the scope of the Lagrangian relaxation approach to problems that do not have a decomposable system action space. Furthermore, it turns out that this alternate Lagrangian relaxation is actually *equivalent* to the ALO: the two models lead to the same bound on the true optimal value function and the same value function approximation (Theorem 2). In contrast, for the classical Lagrangian relaxation, one can only show that the ALO bound is at least as tight [2] and one can find simple examples where the Lagrangian bound can be significantly worse than the ALO bound.

The second way in which our approach builds on the Lagrangian relaxation approach is through its view of time. Our formulation first models the state of each component separately at each decision epoch over a finite horizon before aggregating them over the remaining infinite horizon, whereas the formulation of [2] aggregates them over the entire infinite horizon. While this may appear to be a superficial difference, it turns out to be rather significant because it allows us to prove that the fluid bound is at least as tight as the classical and alternate Lagrangian relaxation bounds (parts (a) and (c) of Theorem 3). As we will see in Section 2.5, the difference in the bounds and the associated performance can be considerable.

With regard to the ALO, the ALO and our fluid model differ in tractability. In particular, the size (number of variables and number of constraints) of our fluid model scales linearly in the number of components and actions. In contrast, in the ALO,

while the number of variables may scale linearly in the number of components, the number of constraints still scales linearly with the number of *system* states, as in the exact LO model of the MDP. This necessitates the use of additional techniques to solve the problem, such as constraint sampling [39]. Moreover, as stated above, it turns out that when one uses a component-wise approximation architecture for the ALO, it is equivalent to the alternate Lagrangian relaxation formulation described above. Due to this equivalence, we are able to assert that our fluid model leads to better bounds than the ALO, and through our numerical results, that our fluid approach leads to better performance than the ALO approach.

Outside of ADP, many approximate approaches to stochastic control problems also exploit decomposability. One salient example of this is the performance region approach to stochastic scheduling. In this approach, one considers a vector of performance measures of the complete system (for an overview, [14]). Using the probabilistic dynamics of the system, one can then derive *conservation laws* that constrain the values this vector of performance measures may take. The resulting set is the performance region of the system, over which one can solve an optimization problem to find the best vector of performance measures. It turns out that typically this vector of performance measures is achieved by simple policies or a randomization of simple policies. This approach was introduced by [35] for multi-class scheduling in a single-server $M/M/1$ queue and later extended by [45] and [88] to more general queueing systems. [18] unified this framework and extended it beyond queueing control problems to such problems as the multiarmed bandit problem and branching bandits. [19] later considered a performance region formulation for the restless bandit problem and used it to derive a high quality heuristic for the restless bandit problem.

Our approach has some conceptual similarities to the performance region approach in the sense that one defines decision variables related to the proportion of time that components of the system are in certain states, imposes constraints that conserve these proportions with each transition and optimizes an objective over the resulting feasible set. In spite of these commonalities, there are a number of key differences. Many existing performance region formulations, due to the nature of the stochastic

system, possess attractive computational and theoretical properties. For example, for systems that satisfy generalized conservation laws, the performance region is an extended polymatroid or contra-polymatroid and so a linear function can be optimized rapidly using a greedy algorithm, and the extreme points of the performance region correspond to deterministic priority rules [18]. In contrast, our formulation does not appear to possess such special computational structure and, as we discuss in Section 2.3.3, optimal solutions of our fluid formulation may in general not be achieved by *any* policy, let alone a specific class of policies. At the same time, many extant performance region approaches are fragile, in that they exploit non-trivial properties of the underlying stochastic system and thus cannot be immediately extended to even simple generalizations. An example of this is the formulation of the multiarmed bandit problem in [18], which exploits specific conservation properties of the regular multiarmed bandit problem and cannot be extended to restless bandits, necessitating the authors' exploration of an alternate approach in [19]. In contrast, our formulation is insensitive to these types of differences; it does not use any structure of the problem beyond the transition probabilities of individual components or groups of components.

Within the performance region literature, our fluid formulation of the restless bandit problem is similar to the performance region model of [19]. This model is actually equivalent to both the classical and the alternate Lagrangian relaxation formulations (Propositions 8 and 9). As a result, our comparison of the fluid formulation with the Lagrangian relaxation formulations allows us to assert that our approach leads to tighter state-wise bounds than the performance region formulation. Moreover, as we will see in Section 2.5.5, our fluid approach significantly outperforms the associated primal dual heuristic of [19].

The first fluid formulation that we will propose in Section 2.3.2 is a countably infinite LO (CILO) problem. There exists a rich literature on this class of problems (see, e.g., [5]). Within this area, the work of [50] and [68] directly studies MDPs; however, both of these papers specifically study *nonstationary* problems and do not additionally consider decomposability. Although we do not explore the application of the methods and theory from the CILO literature to our setting, we believe that it is

an interesting direction for future research.

Lastly, the alternate Lagrangian relaxation we will develop in Section 2.4 arises by relaxing a certain type of action consistency constraint that requires that the action taken in any two components be equivalent. This bears some resemblance to the technique of "variable splitting" or "operator splitting" that is used in continuous optimization (see, for example, [27, 52] and the references therein). The exploration of the connections of the alternate Lagrangian relaxation to splitting-based formulations is an interesting direction for future research.

2.3 Methodology

We begin in Section 2.3.1 by defining a general decomposable, infinite horizon MDP. We then present an infinite LO formulation that is related to this MDP in Section 2.3.2. In Section 2.3.3, we prove some interesting properties of the formulation, and motivated by one of these properties, propose a solvable finite LO formulation and an associated heuristic in Section 2.3.4. The proofs of all theoretical results are provided in Appendix A.1.

2.3.1 Problem definition

In this section, we define the decomposable MDP for which we will subsequently develop our fluid approach.

Let $\mathcal{S}$ be the state space of the complete system, and assume that the complete system state decomposes into M components, so that the complete system state space can be written as $\mathcal{S} = \mathcal{S}^1 \times \cdots \times \mathcal{S}^M$. Let $\mathcal{A}$ be a finite action space, and assume that any action in $\mathcal{A}$ can be taken at any state in $\mathcal{S}$. We make this assumption for simplicity; our approach can be extended to accommodate component-dependent constraints that restrict which actions can be action when specific components enter specific states (for example, an action a cannot be taken when component m is in state k). Let $\mathbf{s}(t)$ be the random variable that represents the state of the complete system at time t, and let $s^m(t)$ denote the state of component m of the system, so

that $\mathbf{s}(t) = (s^1(t), \dots, s^M(t))$. Let $\pi : \{1, 2, \dots\} \times \mathcal{S} \to \mathcal{A}$ be the policy under which the system is operating, which maps a state $\mathbf{s}(t)$ at time t to an action $\pi(t, \mathbf{s}(t))$ in $\mathcal{A}$. Let $p_a(\mathbf{s}, \bar{\mathbf{s}})$ be the probability of the complete system transitioning from state $\mathbf{s}$ to state $\bar{\mathbf{s}}$ in one step when action $a \in \mathcal{A}$ is taken, i.e.,

$$p_a(\mathbf{s}, \bar{\mathbf{s}}) = \mathbb{P}\left(\mathbf{s}(t+1) = \bar{\mathbf{s}} \,|\, \mathbf{s}(t) = \mathbf{s},\ \pi(t, \mathbf{s}(t)) = a\right)$$

for all $t \in \{1, 2, \dots\}$. Let p^m_{kja} denote the probability of component m transitioning from state k to state j in one step when action $a \in \mathcal{A}$ is taken, i.e.,

$$p^m_{kja} = \mathbb{P}(s^m(t+1) = j \,|\, s^m(t) = k,\ \pi(t, \mathbf{s}(t)) = a),$$

for all $t \in \{1, 2, \dots\}$. We assume that the components are independent, so that $p_a(\mathbf{s}, \bar{\mathbf{s}})$ can be written compactly as

$$p_a(\mathbf{s}, \bar{\mathbf{s}}) = \prod_{m=1}^{M} p^m_{s^m \bar{s}^m a}.$$

Let $g_a(\mathbf{s})$ be the reward associated with taking action a when the system is in state $\mathbf{s}$, and assume that it is additive in the components, that is, it can be written as

$$g_a(\mathbf{s}) = \sum_{m=1}^{M} g^m_{s^m a},$$

where g^m_{ka} is the reward associated with taking action a when component m is in state k. Assume that the system starts in state $\mathbf{s} = (s^1, \dots, s^M)$. The problem is then to find a policy π that maximizes the expected total discounted reward:

$$\max_{\pi} \mathbb{E}\left[\sum_{t=1}^{\infty} \sum_{m=1}^{M} \beta^{t-1} g^m_{s^m(t), \pi(t, \mathbf{s}(t))} \,\middle|\, \mathbf{s}(1) = \mathbf{s}\right]. \tag{2.1}$$

2.3.2 Fluid linear optimization formulation

We now consider a fluid formulation of problem (2.1). We begin by defining, for each $t \in \{1, 2, 3 \dots\}$, the decision variable $x^m_{ka}(t)$ to be the proportion of time that

component $m \in \{1, \ldots, M\}$ is in state $k \in \mathcal{S}^m$ and action $a \in \mathcal{A}$ is taken at time t. For every $t \in \{1, 2, 3, \ldots\}$, we also define the decision variable $A_a(t)$ to be the proportion of time that action $a \in \mathcal{A}$ is taken at time t. As stated in Section 2.3.1, our data are the discount factor β, the reward g^m_{ka} (the reward accrued by the decision maker when component m is in state k and action a is taken) and the transition probability p^m_{kja} (the probability that component m transitions from state k to state j in one step when action a is taken).

Finally, we assume that the system starts deterministically in a state $\mathbf{s} \in \mathcal{S}$. For convenience, we will define $\alpha^m_k(\mathbf{s})$ as

$$\alpha^m_k(\mathbf{s}) = \begin{cases} 1, & \text{if } s^m = k, \\ 0, & \text{otherwise.} \end{cases}$$

The fluid problem for initial state $\mathbf{s}$ can now be formulated as follows:

$$\underset{\mathbf{x},\mathbf{A}}{\text{maximize}} \quad \sum_{t=1}^{\infty} \sum_{m=1}^{M} \sum_{k \in \mathcal{S}^m} \sum_{a \in \mathcal{A}} \beta^{t-1} \cdot g^m_{ka} \cdot x^m_{ka}(t) \tag{2.2a}$$

$$\text{subject to} \quad \sum_{a \in \mathcal{A}} x^m_{ja}(t) = \sum_{k \in \mathcal{S}^m} \sum_{\tilde{a} \in \mathcal{A}} p^m_{kj\tilde{a}} x^m_{k\tilde{a}}(t-1),$$

$$\forall\ m \in \{1, \ldots, M\},\ t \in \{2, 3, \ldots\},\ j \in \mathcal{S}^m, \tag{2.2b}$$

$$\sum_{k \in \mathcal{S}^m} x^m_{ka}(t) = A_a(t), \qquad \forall\ m \in \{1, \ldots, M\},\ a \in \mathcal{A},\ t \in \{1, 2, \ldots\}, \tag{2.2c}$$

$$\sum_{a \in \mathcal{A}} x^m_{ka}(1) = \alpha^m_k(\mathbf{s}), \qquad \forall\ m \in \{1, \ldots, M\},\ k \in \mathcal{S}^m, \tag{2.2d}$$

$$x^m_{ka}(t) \geq 0, \qquad \forall\ m \in \{1, \ldots, M\},\ a \in \mathcal{A},\ k \in \mathcal{S}^m,\ t \in \{1, 2, \ldots\}, \tag{2.2e}$$

$$A_a(t) \geq 0, \qquad \forall\ a \in \mathcal{A},\ t \in \{1, 2, \ldots\}. \tag{2.2f}$$

Constraint (2.2b) ensures that probability is conserved from time $t-1$ to time t: the left hand side represents the proportion of time that component m is in state j at time t in terms of the $x^m_{ja}(t)$ variables, while the right hand side represents the

same proportion, only in terms of the $x^m_{k\tilde{a}}(t-1)$ variables, which correspond to time $t-1$. Constraint (2.2c) ensures that, for each component, the proportion of time that action a is taken in terms of the $x^m_{ka}(t)$ variables is equal to $A_a(t)$ (which is precisely defined as the proportion of time that action a is taken at time t). Thus, the actions $a \in \mathcal{A}$ connect the variables corresponding to the different components. Constraint (2.2d) ensures that the initial frequency with which each component m is in a state k is exactly $\alpha^m_k(\mathbf{s})$. The remaining two constraints (2.2e) and (2.2f) ensure that all of the decision variables are nonnegative, as they represent proportions. Given the definition of $x^m_{ka}(t)$ as the proportion of time that component m is in state k at time t and action a is taken at time t, the objective can therefore be interpreted as the expected discounted long term reward.

Note that constraints (2.2b) and (2.2d), together with the fact that $\sum_{j\in\mathcal{S}^m} p^m_{kja} = 1$ for any m, k and a, imply that $\sum_{k\in\mathcal{S}^m}\sum_{a\in\mathcal{A}} x^m_{ka}(t) = 1$ for each m and t. Together with constraint (2.2c), this also implies that $\sum_{a\in\mathcal{A}} A_a(t) = 1$ for each t.

The following result, which follows by standard arguments in infinite dimensional linear optimization (see Section A.1.1 in Appendix A), establishes that the infinite horizon problem (2.2) is well-defined.

Proposition 1 *For each $\mathbf{s} \in \mathcal{S}$, problem (2.2) has an optimal solution.*

2.3.3 Properties of the infinite fluid LO

We will now develop some theoretical properties of the fluid LO model. Let $(\mathbf{x}(\mathbf{s}), \mathbf{A}(\mathbf{s}))$ and $Z^*(\mathbf{s})$ denote an optimal solution and the optimal objective value, respectively, to problem (2.2) corresponding to initial state $\mathbf{s}$. Denote by $J^*(\cdot)$ the optimal value function obtained using dynamic programming, that is,

$$J^*(\mathbf{s}) = \max_{\pi} \mathbb{E}\left[\sum_{t=1}^{\infty}\sum_{m=1}^{M} \beta^{t-1} g^m_{s^m(t),\pi(t,\mathbf{s}(t))} \,\middle|\, \mathbf{s}(1) = \mathbf{s}\right]$$

for every $\mathbf{s} \in \mathcal{S}$. We then have the following relationship between problem (2.2) and the optimal value function, whose proof appears as Section A.1.2 in Appendix A.

Proposition 2 *For every* $\mathbf{s} \in \mathcal{S}$, $J^*(\mathbf{s}) \leq Z^*(\mathbf{s})$.

The idea behind the proof of Proposition 2, is that by using an optimal policy π^*, it is possible to construct a feasible solution $(\mathbf{x}, \mathbf{A})$ to problem (2.2) whose objective value is the true optimal value $J^*(\mathbf{s})$. Unfortunately, the opposite inequality does not hold in general; see Appendix A.2 counterexample. Let us call the optimal solution $(\mathbf{x}(\mathbf{s}), \mathbf{A}(\mathbf{s}))$ *achievable* if there exists a (possibly non-deterministic and time-varying) policy π such that

$$\begin{aligned} x_{ka}^m(t, \mathbf{s}) &= \mathbb{P}(s^m(t) = k,\ \pi(t, \mathbf{s}(t)) = a), && \forall\ m \in \{1, \ldots, M\},\ k \in \mathcal{S}^m, \\ & && a \in \mathcal{A},\ t \in \{1, 2, \ldots\}, \\ A_a(t, \mathbf{s}) &= \mathbb{P}(\pi(t, \mathbf{s}(t)) = a), && \forall\ a \in \mathcal{A},\ t \in \{1, 2, \ldots\}, \end{aligned}$$

where $\mathbf{s}(t)$ is the state of the complete system stochastic process at time t, operated according to π, starting from $\mathbf{s}$ (i.e., $\mathbf{s}(1) = \mathbf{s}$). (Note that we use $x_{ka}^m(t, \mathbf{s})$ and $A_a(t, \mathbf{s})$ to denote the optimal value of $x_{ka}^m(t)$ and $A_a(t)$ in the solution $(\mathbf{x}(\mathbf{s}), \mathbf{A}(\mathbf{s}))$ that corresponds to initial state $\mathbf{s}$.) Under the assumption of achievability, we have the following result.

Proposition 3 *Let* $\mathbf{s} \in \mathcal{S}$. *If* $(\mathbf{x}(\mathbf{s}), \mathbf{A}(\mathbf{s}))$ *is achievable, then* $Z^*(\mathbf{s}) \leq J^*(\mathbf{s})$.

The proof is contained Section A.1.3 of Appendix A. The result follows since, under the assumption of achievability, $Z^*(\mathbf{s})$ is the total expected discounted reward of some policy, while $J^*(\mathbf{s})$ is the highest any such reward can be.

Under the assumptions of component independence and achievability, the fluid formulation allows us to construct an optimal policy.

Theorem 1 *Suppose that for all* $\mathbf{s} \in \mathcal{S}$, $(\mathbf{x}(\mathbf{s}), \mathbf{A}(\mathbf{s}))$ *is achievable. Define the deterministic, stationary policy* $\pi : \mathcal{S} \to \mathcal{A}$ *as*

$$\pi(\mathbf{s}) = \arg\max_{a \in \mathcal{A}} A_a(1, \mathbf{s}).$$

Under these assumptions, the policy π *is an optimal policy, i.e.,* π *solves problem* (2.1).

The proof of the result (found in Section A.1.4 of Appendix A) follows by showing that any action a such that $A_a(1, \mathbf{s}) > 0$ is an action that is greedy with respect to the objective value $Z^*(\cdot)$ which, by combining Propositions 2 and 3, is equal to the optimal value function $J^*(\cdot)$.

2.3.4 Fluid-based heuristic

Theorem 1 tells us that, assuming that for every initial state $\mathbf{s}$ the optimal solution of problem (2.2) for initial state $\mathbf{s}$ is achievable, we immediately have an optimal policy by simply looking at the optimal values of the A variables at the first period ($t = 1$). Typically, however, the optimal solution of (2.2) will not be achievable. It nevertheless seems reasonable to expect that in many problems, the optimal solution $(\mathbf{x}(\mathbf{s}), \mathbf{A}(\mathbf{s}))$ may be close to being achievable for many states $\mathbf{s}$, because $(\mathbf{x}(\mathbf{s}), \mathbf{A}(\mathbf{s}))$ still respects the transition behavior of the system at the level of individual components. Consequently, the action $\arg\max_{a \in \mathcal{A}} A_a(1, \mathbf{s})$ should then be close to an optimal action for many states $\mathbf{s}$. It is therefore reasonable to expect that, by selecting the action a as $\arg\max_{a \in \mathcal{A}} A_a(1, \mathbf{s})$, one may often still be able to get good performance, even though the optimal solution of problem (2.2) may not be achievable.

Notwithstanding the question of achievability, applying this intuition in practice is not immediately possible. The reason for this is that problem (2.2) is an LO problem with a countably infinite number of variables and constraints, and so cannot be solved using standard solvers. Towards the goal of developing a practical heuristic policy for problem (2.1), we now consider an alternate, finite problem that can be viewed as an approximation to problem (2.2). This new formulation, presented below as problem (2.3), requires the decision maker to specify a time horizon T over which the evolution of the system will be modeled. For $t \in \{1, \ldots, T\}$, the variables $x_{ka}^m(t)$ and $A_a(t)$ have the same meaning as in problem (2.3). To model the evolution of the system beyond $t = T$, we use the variable $x_{ka}^m(T + 1)$ to represent the expected discounted long-run frequency with which component m is in state k and action a is taken from $t = T + 1$ on. Similarly, we use $A_a(T + 1)$ to represent the expected discounted frequency with which action a is taken from $t = T + 1$ on.

With these definitions, the formulation corresponding to initial state $\mathbf{s}$ is presented below.

$$\underset{\mathbf{x},\mathbf{A}}{\text{maximize}} \quad \sum_{t=1}^{T+1} \sum_{m=1}^{M} \sum_{k \in \mathcal{S}^m} \sum_{a \in \mathcal{A}} \beta^{t-1} \cdot g_{ka}^m \cdot x_{ka}^m(t) \tag{2.3a}$$

$$\text{subject to} \quad \sum_{a \in \mathcal{A}} x_{ja}^m(t) = \sum_{k \in \mathcal{S}^m} \sum_{\tilde{a} \in \mathcal{A}} p_{kj\tilde{a}}^m \cdot x_{k\tilde{a}}^m(t-1),$$

$$\forall\ m \in \{1, \ldots, M\},\ t \in \{2, \ldots, T\},\ j \in \mathcal{S}^m, \tag{2.3b}$$

$$\sum_{a \in \mathcal{A}} x_{ja}^m(T+1) = \sum_{k \in \mathcal{S}^m} \sum_{a \in \mathcal{A}} p_{kja}^m \cdot x_{ka}^m(T) + \beta \cdot \sum_{k \in \mathcal{S}^m} \sum_{a \in \mathcal{A}} p_{kja}^m \cdot x_{ka}^m(T+1),$$

$$\forall\ m \in \{1, \ldots, M\},\ j \in \mathcal{S}^m, \tag{2.3c}$$

$$\sum_{k \in \mathcal{S}^m} x_{ka}^m(t) = A_a(t), \qquad \forall\ m \in \{1, \ldots, M\},\ a \in \mathcal{A},\ t \in \{1, \ldots, T+1\} \tag{2.3d}$$

$$\sum_{a \in \mathcal{A}} x_{ka}^m(1) = \alpha_k^m(\mathbf{s}), \qquad \forall\ m \in \{1, \ldots, M\},\ k \in \mathcal{S}^m, \tag{2.3e}$$

$$x_{ka}^m(t) \geq 0, \qquad \forall\ m \in \{1, \ldots, M\},\ a \in \mathcal{A},\ k \in \mathcal{S}^m,\ t \in \{1, \ldots, T+1\}, \tag{2.3f}$$

$$A_a(t) \geq 0, \qquad \forall\ a \in \mathcal{A},\ t \in \{1, \ldots, T+1\}. \tag{2.3g}$$

With regard to constraints, we retain the same conservation constraints that relate the x_{ka}^m variables at $t-1$ to t, the initial state constraint and the consistency constraints that relate the x_{ka}^m and the A_a variables at a time t, for $t \in \{1, \ldots, T\}$. Beyond $t = T$, constraint (2.3c) models the long-run transition behavior of the system. This constraint can be interpreted as a conservation relation: the left hand side represents the expected discounted number of times from $T+1$ on that we take an action out of component m being in state j, while the right hand side represents the expected discounted number of times that we enter state j from $T+1$ on. More specifically, the first right-hand side term represents the expected number of times that we enter state j at time $T+1$ (which is not discounted, since $T+1$ is the first period of the horizon $\{T+1, T+2, T+3, \ldots\}$) and the second term represents the expected discounted

number of times that we enter state j from $T+2$ on. Note also that constraint (2.3d), which is the analog of constraint (2.2c), extends from $t = 1$ to $t = T+1$, ensuring that the $x^m_{ka}(T+1)$ and the $A_a(T+1)$ variables are also consistent with each other. With regard to the objective, observe that rather than being an infinite sum from $t = 1$, the objective of problem (2.3) is a finite sum that extends from $t = 1$ to $t = T+1$.

Let $Z^*_T(\mathbf{s})$ denote the optimal value of problem (2.3). Problem (2.3), like problem (2.2), provides an upper bound on the optimal value function at $J^*(\mathbf{s})$, and this bound improves with T, as indicated by the following result.

Proposition 4 *For each* $\mathbf{s} \in \mathcal{S}$ *and all* $T \in \{1, 2, \dots\}$*:*

(a) $Z^*_T(\mathbf{s}) \geq J^*(\mathbf{s})$*; and*

(b) $Z^*_T(\mathbf{s}) \geq Z^*_{T+1}(\mathbf{s})$.

The proof of part (a) of Proposition 4 follows along similar lines to Proposition 2, while the proof of part (b) follows by showing that a solution to problem (2.3) with $T+1$ can be used to construct a feasible solution for problem (2.3) with T that achieves an objective value of $Z^*_{T+1}(\mathbf{s})$. The proof of this proposition can be found in Section A.1.5 of Appendix A. Part (a) of the proposition is useful because in passing from the infinite to the finite formulation, we have not lost the useful property that the objective value provides an upper bound on the optimal value function. Part (b) is important because it suggests a tradeoff in bound quality and computation: by increasing T, the quality of the bound improves, but the size of the formulation (the number of variables and constraints) increases. We will see later in Sections 2.5.4 and 2.5.5 that typically T does not need to be very large to ensure strong bounds and performance.

With this formulation, our heuristic policy is then defined as Algorithm 1.

Before continuing, we comment on two important ways in which problem (2.3) can be extended and one limitation of formulation (2.3). First of all, in problem (2.3), we formulated the decomposable MDP problem by defining decision variables that correspond to first-order information: in particular, $x^m_{ka}(t)$ represents the frequency

Algorithm 1 Fluid LO heuristic for infinite horizon problem with known stationary probabilities.

Require: Parameter T; data $\mathbf{p}$, $\mathbf{g}$, β; current state $\mathbf{s} \in \mathcal{S}$.

Solve problem (2.3) corresponding to initial state $\mathbf{s}$, horizon T and data $\mathbf{p}$, $\mathbf{g}$, β to obtain an optimal solution $(\mathbf{x}(\mathbf{s}), \mathbf{A}(\mathbf{s}))$.

Take action $\tilde{a}$, where $\tilde{a} = \arg\max_{a \in \mathcal{A}} A_a(1, \mathbf{s})$.

with which a *single* component (component m) is in state k and action a is taken at time t. As shown in Section 2.3.3, the resulting formulation provides an upper bound on the optimal expected discounted reward. We can improve on this by considering higher-order fluid formulations, where rather than defining our decision variables to correspond to one component being in a state, we can define decision variables corresponding to combinations of components being in combinations of states, while a certain action is taken at a certain time. For example, a second-order formulation would correspond to using decision variables that model how frequently *pairs* of components are in different *pairs* of states while an action is taken at each time. As the order of the formulation increases, the objective value becomes an increasingly tighter bound on the optimal value, and it may be reasonable to expect better performance from using Algorithm 1; however, the size of the formulation increases rapidly.

Second, problem (2.3) models an infinite horizon problem and Algorithm 1 is a heuristic for this problem. For finite horizon problems, we can apply our approach as follows. Problem (2.3) can be modified by setting T to the horizon of the actual problem and removing the terminal $T+1$ decision variables that model the long-run evolution of the system. Then, if we are at state $\mathbf{s}$ at period t', we restrict the fluid problem to $\{t', t'+1, \ldots, T\}$ and use constraint (2.3e) to set the initial state at t' to $\mathbf{s}$. We then solve the problem to obtain the optimal solution $(\mathbf{x}(\mathbf{s}), \mathbf{A}(\mathbf{s}))$ and we take the action a that maximizes $A_a(t, \mathbf{s})$. Note that if the transition probabilities change over time (i.e., rather than p^m_{kja} we have $p^m_{kja}(t)$ for $t \in \{1, \ldots, T-1\}$), we may also modify constraint (2.3b) and replace p^m_{kja} with $p^m_{kja}(t)$, without changing the size or the nature of the resulting formulation.

Finally, we comment on one limitation to the fluid formulation (2.3). Problem (2.3) is formulated in terms of the *system* action space $\mathcal{A}$; the actions that index

the $x^m_{ka}(t)$ and $A_a(t)$ variables are elements of the system action space $\mathcal{A}$. For certain problems, the system action space $\mathcal{A}$ may be small and problem (2.3) may be easy to solve. For example, in a multiarmed bandit problem where exactly one bandit must be activated, $|\mathcal{A}| = M$ (one of the M bandits); similarly, in an optimal stopping problem, $|\mathcal{A}| = 2$ (stop or continue). For other problems, the action space of the problem may grow exponentially (e.g., a bandit problem where one may activate up to K of M bandits). For such problems, the fluid formulation (2.3) will be harder to solve; we do not consider this regime here. The development of a scalable solution method for large scale versions of problem (2.3) constitutes an interesting direction for future research.

2.4 Comparisons to other approaches

In this section, we compare our finite fluid formulation (2.3) against three state-of-the-art formulations that can be used to solve decomposable MDPs. We begin by stating these formulations: in Sections 2.4.1, 2.4.2 and 2.4.3, we present the ALO, classical Lagrangian relaxation and the alternate Lagrangian relaxation formulations, respectively. Then, in Section 2.4.4 we state a key theoretical result that asserts that the finite fluid formulation (2.3) provides a provably tighter bound than all three formulations. Finally, we conclude with a discussion of the sizes of the formulations in Section 2.4.5.

2.4.1 Approximate linear optimization

For the ALO formulation of [38], we approximate the value function using the same functional form as in [2]:

$$J_{ALO}(\mathbf{s}) = \sum_{m=1}^{M} J^m_{s^m}, \tag{2.4}$$

i.e., we assume that each state of each component contributes an additive effect. For a given initial state $\mathbf{s} \in \mathcal{S}$, the corresponding ALO formulation is then

$$\underset{\mathbf{J}}{\text{minimize}} \quad \sum_{m=1}^{M} \sum_{k \in \mathcal{S}^m} \alpha_k^m(\mathbf{s}) \cdot J_k^m \tag{2.5a}$$

$$\text{subject to} \quad \sum_{m=1}^{M} J_{\bar{s}^m}^m \geq \sum_{m=1}^{M} g_{\bar{s}^m a}^m + \beta \sum_{m=1}^{M} \sum_{j \in \mathcal{S}^m} p_{\bar{s}^m j a}^m J_j^m, \quad \forall\ \bar{\mathbf{s}} \in \mathcal{S},\ a \in \mathcal{A}. \tag{2.5b}$$

To derive a policy from $\mathbf{J}$, we take the action $\tilde{a}$ that is greedy with respect to J_{ALO}; this action is defined as

$$\tilde{a} = \arg\max_{a \in \mathcal{A}} \left\{ \sum_{m=1}^{M} g_{s^m a}^m + \beta \cdot \sum_{m=1}^{M} \sum_{j \in \mathcal{S}^m} p_{s^m j a}^m J_j^m \right\}. \tag{2.6}$$

Let $Z_{ALO}^*(\mathbf{s})$ denote the objective value of problem (2.5) with initial state $\mathbf{s}$. The following result, due to [2], establishes that $Z_{ALO}^*(\mathbf{s})$ upper bounds the optimal value function at $\mathbf{s}$. The proof can be found in [2] and is thus omitted.

Proposition 5 *(Proposition 4 of [2].) For all* $\mathbf{s} \in \mathcal{S}$, $Z_{ALO}^*(\mathbf{s}) \geq J^*(\mathbf{s})$.

2.4.2 Classical Lagrangian relaxation

We now present the classical Lagrangian relaxation (CLR) approach. In order to apply this approach to our decomposable MDP defined in Section 2.3.1, we require three additional assumptions.

Assumption 1 *In addition to the system state space being decomposable along components, the action space also decomposes along the components. More precisely, each component* m *is endowed with both a state space* $\mathcal{S}^m$ *and an action space* $\mathcal{A}^m$*. Thus, an action* a *in the system action space can be represented as a tuple of component actions,* $a = (a^1, \ldots, a^m) \in \mathcal{A} \subseteq \mathcal{A}^1 \times \cdots \times \mathcal{A}^M$.

Assumption 2 *The rewards and transition probabilities decompose with respect to the new action spaces* $\mathcal{A}^m$*. Let* $R_{ka^m}^m$ *denote the reward from component* m *when action*

a^m is taken in state k and let $\bar{p}^m_{ka^m}$ denote the transition probability of component m when action a^m is taken. We require that $p^m_{kja} = \bar{p}^m_{ka^m}$ and $g^m_{ka} = R^m_{ka^m}$ whenever the mth component of a is a^m.

Assumption 3 *The system action state space $\mathcal{A}$ is defined implicitly through a linking constraint on the component actions:*

$$\mathcal{A} = \left\{ a = (a^1, \ldots, a^M) \in \mathcal{A}^1 \times \cdots \times \mathcal{A}^M \;\middle|\; \sum_{m=1}^{M} \mathbf{D}^m(a^m) \leq \mathbf{b} \right\}, \tag{2.7}$$

where $\mathbf{D}^m : \mathcal{A}^m \to \mathbb{R}^q$ is a function for each m and $\mathbf{b} \in \mathbb{R}^q$ for some finite q.

When these three assumptions hold, the Lagrangian approach involves dualizing the linking constraint $\sum_{m=1}^{M} \mathbf{D}^m(a^m) \leq \mathbf{b}$ by introducing a Lagrange multiplier vector $\boldsymbol{\lambda} \in \mathbb{R}^p$ for this linking constraint. The CLR formulation of the problem can be written as follows:

$$\begin{aligned}
\underset{\boldsymbol{\lambda}, \mathbf{V}}{\text{minimize}} \quad & \frac{\boldsymbol{\lambda}^T \mathbf{b}}{1-\beta} + \sum_{m=1}^{M} \sum_{k \in \mathcal{S}^m} \alpha^m_k(\mathbf{s}) \cdot V^m_k && \text{(2.8a)} \\
\text{subject to} \quad & V^m_k \geq R^m_{ka^m} - \boldsymbol{\lambda}^T \mathbf{D}^m(a^m) + \beta \cdot \sum_{j \in \mathcal{S}^m} \bar{p}^m_{kja^m} \cdot V^m_j, \\
& \qquad \forall\, m \in \{1, \ldots, M\},\ k \in \mathcal{S}^m,\ a^m \in \mathcal{A}^m && \text{(2.8b)} \\
& \boldsymbol{\lambda} \geq 0. && \text{(2.8c)}
\end{aligned}$$

The optimal variable $\mathbf{V}$ can be interpreted as a component-wise approximation to the value function. One can form a value function approximation that is analogous to the ALO approximation in equation (2.4) and take the greedy action analogously to equation (2.6).

The dual of problem (2.8) is

$$\underset{\mathbf{z}}{\text{maximize}} \quad \sum_{m=1}^{M} \sum_{k \in \mathcal{S}^m} \sum_{a^m \in \mathcal{A}^m} R^m_{ka^m} \cdot z^m_{ka^m} \tag{2.9a}$$

$$\text{subject to} \quad \sum_{a^m \in \mathcal{A}^m} z^m_{ja^m} = \alpha^m_j(\mathbf{s}) + \beta \cdot \sum_{k \in \mathcal{S}^m} \sum_{a^m \in \mathcal{A}^m} \bar{p}^m_{kja^m} \cdot z^m_{ka^m},$$

$$\forall\ m \in \{1, \ldots, M\},\ j \in \mathcal{S}^m, \tag{2.9b}$$

$$\sum_{m=1}^{M} \sum_{k \in \mathcal{S}^m} \sum_{a^m \in \mathcal{A}^m} \mathbf{D}^m(a^m) z^m_{ka^m} \leq \frac{\mathbf{b}}{1-\beta}, \tag{2.9c}$$

$$z^m_{ka^m} \geq 0, \quad \forall\ m \in \{1, \ldots, M\},\ k \in \mathcal{S}^m,\ a^m \in \mathcal{A}^m. \tag{2.9d}$$

The variable $z^m_{ka^m}$ can be interpreted as the expected discounted frequency with which component m is in state k and action a^m is being taken over the entire infinite horizon. Constraint (2.9b) models the transition dynamics of component m in an expected discounted sense, while constraint (2.9c) can be interpreted as the expected discounted version of the linking constraint that defines the action space $\mathcal{A}$ in equation (2.7).

Let $Z^*_{CLR}(\mathbf{s})$ be the optimal objective value of problem (2.8) corresponding to initial state $\mathbf{s}$. The following two results, due to [2], establish that the CLR provides an upper bound on the optimal value function and that ALO provides a tighter bound than the CLR. The proofs can be found in [2] and are omitted.

Proposition 6 *(Proposition 2 of [2].) When Assumptions 1 – 3 hold, for all* $\mathbf{s} \in \mathcal{S}$, $Z^*_{CLR}(\mathbf{s}) \geq J^*(\mathbf{s})$.

Proposition 7 *(Corollary 1 of [2].) When Assumptions 1 – 3 hold, for all* $\mathbf{s} \in \mathcal{S}$, $Z^*_{ALO}(\mathbf{s}) \leq Z^*_{CLR}(\mathbf{s})$.

Furthermore, [2] provide a simple parameterized problem (Section 3.3 of that paper) where the difference between $Z^*_{ALO}(\mathbf{s})$ and $Z^*_{CLR}(\mathbf{s})$ can be made arbitrarily large.

2.4.3 Alternate Lagrangian relaxation

The CLR formulation requires Assumptions 1 – 3 to hold. When these assumptions hold, it is possible to exploit the definition of the system action space in equation (2.7)

in order to arrive at formulation (2.8). However, the decomposable MDP that we have defined in Section 2.3.1 may not be consistent with these assumptions; more precisely, the system action space may not naturally decompose along the components. Consider, for example, an optimal stopping problem where the system is actually M independent components. In this example, the action space (which consists of two actions, stop or continue) does not decompose along each component, and it does not make sense to think of the system action space as being the feasible set of a coupling constraint on the action spaces of M small MDPs.

Surprisingly, it turns out that there is a transformation by which one can convert any decomposable MDP with a general system action space $\mathcal{A}$ into a weakly coupled MDP and thus apply the Lagrangian relaxation approach, even when the action space $\mathcal{A}$ does not have a representation of the form in equation (2.7). The steps of this transformation are as follows.

1. Construct M small MDPs, where the mth small MDP corresponds to component m of the decomposable MDP.

2. Set the state space of small MDP m to be $\mathcal{S}^m$, the state space of component m.

3. Set the action space of small MDP m to be $\mathcal{A}$, the action space of the complete system. (Thus, each small MDP involves controlling how component m evolves across its own state space, where we may choose any action from the *system* action space $\mathcal{A}$.)

4. Enforce the following coupling constraint:

$$\mathbb{I}\{a^m = a\} - \mathbb{I}\{a^{m+1} = a\} = 0, \quad \forall\, m \in \{1, \ldots, M-1\},\ a \in \mathcal{A}, \tag{2.10}$$

or equivalently, that

$$a^m = a^{m+1}, \quad \forall\, m \in \{1, \ldots, M-1\}.$$

The above constraint is simple: it requires that the actions taken in small MDP

m and small MDP $m+1$ must be the same, or equivalently, the actions $a^m, a^{m'}$ taken in any pair of small MDPs $m, m' \in \{1, \ldots, M\}$ must be the same.

It is easy to see that this weakly coupled MDP is exactly the same as the decomposable MDP of Section 2.3.1. We now construct the Lagrangian relaxation of this weakly coupled MDP. Introducing the Lagrange multiplier λ_a^m for the (m, a) constraint in the family of constraints (2.10) and using $\boldsymbol{\lambda}$ to denote the vector of multipliers, the corresponding Lagrangian relaxation formulation of this weakly coupled MDP for initial state $\mathbf{s}$, can be shown to be

$$\underset{\boldsymbol{\lambda}, \mathbf{V}}{\text{minimize}} \quad \sum_{m=1}^{M} \sum_{k \in \mathcal{S}^m} \alpha_k^m(\mathbf{s}) V_k^m \tag{2.11a}$$

$$\text{subject to} \quad V_k^m \geq g_{ka}^m - \mathbb{I}\{m < M\} \cdot \lambda_a^m + \mathbb{I}\{m > 1\} \cdot \lambda_a^{m-1} + \beta \cdot \sum_{j \in \mathcal{S}^m} p_{kja}^m V_j^m,$$

$$\forall\ m \in \{1, \ldots, M\},\ k \in \mathcal{S}^m,\ a \in \mathcal{A}. \tag{2.11b}$$

We refer to this relaxation as the *alternate Lagrangian relaxation* (ALR). A more detailed derivation of the ALR can be found in Appendix A.3. As with the CLR and the ALO, one can form a value function approximation of the form in equation (2.4) using the optimal $\mathbf{V}$ values and take the greedy action analogously to equation (2.6).

It should be clear that the ALR problem (2.11) is *not* the same as the CLR problem (2.8). Problem (2.11) only decomposes the state and does not decompose the system action space; it accomplishes this by endowing each component with the system action space and enforcing the action consistency constraint (2.10). Problem (2.8), on the other hand, decomposes the state *and* the action space by using the structure of the action space given in equation (2.7). The resulting formulations thus differ in their sizes; typically, the ALR will be larger than the CLR because the dimensions of the ALR problem (numbers of variables and constraints) scale with the number of *system* actions. Note that problem (2.8) can be formulated only when Assumptions 1 – 3 hold. On the other hand, problem (2.11) can *always* be formulated, regardless of the structure of the action space. To the best of our knowledge, this type of alternate Lagrangian relaxation has not been proposed before.

To understand how the ALR relates to the fluid formulation, it is helpful to formulate the dual of problem (2.11):

$$\underset{\mathbf{z}}{\text{maximize}} \quad \sum_{m=1}^{M} \sum_{k \in \mathcal{S}^m} \sum_{a \in \mathcal{A}} g_{ka}^m z_{ka}^m \tag{2.12a}$$

$$\text{subject to} \quad \sum_{a' \in \mathcal{A}} z_{ja'}^m - \beta \sum_{k \in \mathcal{S}^m} \sum_{a \in \mathcal{A}} p_{kja}^m \cdot z_{ka}^m = \alpha_j^m(\mathbf{s}), \quad \forall\ m \in \{1, \ldots, M\},\ j \in \mathcal{S}^m, \tag{2.12b}$$

$$\sum_{k \in \mathcal{S}^m} z_{ka}^m = \sum_{k \in \mathcal{S}^{m+1}} z_{ka}^{m+1}, \quad \forall\ m \in \{1, \ldots, M-1\},\ a \in \mathcal{A}, \tag{2.12c}$$

$$z_{ka}^m \geq 0, \quad \forall\ m \in \{1, \ldots, M\},\ k \in \mathcal{S}^m,\ a \in \mathcal{A}. \tag{2.12d}$$

The dual variable z_{ka}^m can be interpreted as the expected discounted number of times that the component m is in state k and action a is taken over the entire horizon. Constraint (2.12b) models the long-term transition behavior of small MDP m, while constraint (2.12c) can be interpreted as the expected discounted version of the linking constraint (2.10); the expected discounted number of times that we take action a in small MDP m must be the same as the expected discounted number of times that we take action a in small MDP $m+1$.

Having formed the dual problem (2.12), we can see that the ALR dual (2.12) and the finite fluid formulation (2.3) bear some resemblance, in terms of accounting for how frequently components are in specific states while a specific action is taken, accounting for the transition behavior and accounting for the fact that component state-action frequencies (the $x_{ka}^m(t)$ variables in problem (2.3) and the z_{ka}^m variables in problem (2.12)) are linked across components through the action. However, the key difference lies in the fact that in problem (2.12), time is fully aggregated: the z_{ka}^m variables represent the *long run* expected discounted frequency with which component m is in state k and action a is taken from $t = 1$ on. In contrast, in problem (2.3), time is partially disaggregated: for $t = 1$ to $t = T$, the transition behavior of the system is modeled separately for each t, and for $t = T+1$ and beyond, the transition behavior is modeled in the same aggregate sense (compare constraints (2.12b) and

(2.3c)). One can thus interpret the fluid formulation as a partially disaggregated version of the ALR dual (2.12). If one imagines the finite fluid formulation (2.3) with $T = 0$ – i.e., the formulation does not account for the transition behavior separately for any periods and only accounts for transition behavior in a long-term discounted sense, from period 1 ($= T + 1$) on – then one can see that it would be equivalent to the ALR dual problem (2.12).

Let $Z^*_{ALR}(\mathbf{s})$ denote the optimal value of the ALR formulation (2.11) corresponding to initial state $\mathbf{s}$ and let $Z^*_{ALO}(\mathbf{s})$ denote the optimal value of the ALO formulation (2.5) corresponding to initial state $\mathbf{s}$. The following result establishes that problems (2.11) and (2.5) are equivalent.

Theorem 2 *For each* $\mathbf{s} \in \mathcal{S}$*:*

(a) $Z^*_{ALO}(\mathbf{s}) = Z^*_{ALR}(\mathbf{s})$*; and*

(b) Let $\mathbf{V} \in \mathbb{R}^{\sum_{m=1}^{M} |\mathcal{S}^m|}$*. There exists* $\boldsymbol{\lambda}$ *such that* $(\mathbf{V}, \boldsymbol{\lambda})$ *is an optimal solution for the Lagrangian relaxation formulation* (2.11) *corresponding to state* $\mathbf{s}$ *if and only if* $\mathbf{V}$ *is an optimal solution for the ALO formulation* (2.5) *corresponding to state* $\mathbf{s}$*.*

The proof of this result, found in Section A.1.6 of Appendix A, follows by essentially showing that the optimal solution of one problem leads to a feasible solution for the other problem with the same optimal value.

We offer two remarks on Theorem 2. First, we believe Theorem 2 to be valuable because the alternate Lagrangian relaxation problem (2.11) is considerably more tractable than the ALO problem (2.5). Specifically, the former has a number of variables and constraints that is linear in the problem dimensions, while the latter has a number of constraints that is in general exponential in the number of components. One of the challenges of applying the ALO approach is that, although applying a basis function approximation as in equation (2.4) allows one to reduce the number of *variables*, one is still left with a large number of *constraints* (one for each pair $(\mathbf{s}, a) \in \mathcal{S} \times \mathcal{A}$). To cope with the large number of constraints, one might use constraint sampling [39] or column generation techniques (see, e.g., [1, 2]). Theorem 2

implies that in cases where $\mathcal{A}$ is not too large, one can avoid resorting to these techniques by directly solving problem (2.11): by part (a) of the theorem, the resulting bound will be the same as that of the ALO formulation, and by part (b), the resulting value function approximation is also a valid ALO value function approximation (since the optimal $\mathbf{V}$ for the ALR problem (2.11) is also a valid optimal solution for the ALO problem (2.5)).

Second, it is valuable to contrast Theorem 2 to Proposition 7, which pertains to the relationship between the CLR and the ALO formulations. Theorem 2 asserts that the ALR and the ALO are equivalent, whereas Proposition 7 asserts that the CLR is no tighter than the ALO (moreover, as discussed in Section 2.4.2 there exist simple examples where the difference between $Z^*_{ALO}(\mathbf{s})$ and $Z^*_{CLR}(\mathbf{s})$ can be extremely large). Thus, problem (2.11) is a tighter formulation of the MDP than problem (2.8), as summarized in the following corollary.

Corollary 1 *When Assumptions 1 – 3 hold, for all* $\mathbf{s} \in \mathcal{S}$, $Z^*_{ALR}(\mathbf{s}) \leq Z^*_{CLR}(\mathbf{s})$.

2.4.4 Comparison of bounds

With Theorem 2 in hand, we are ready to state our key theoretical result, whose proof appears in Section A.1.7 of Appendix A.

Theorem 3 *For each* $\mathbf{s} \in \mathcal{S}$ *and all* $T \in \{1, 2, \dots\}$*:*

(a) $Z^*_T(\mathbf{s}) \leq Z^*_{ALR}(\mathbf{s})$*;*

(b) $Z^*_T(\mathbf{s}) \leq Z^*_{ALO}(\mathbf{s})$*; and*

(c) $Z^*_T(\mathbf{s}) \leq Z^*_{CLR}(\mathbf{s})$ *(when Assumptions 1 – 3 hold).*

Part (a) follows by using a solution of problem (2.3) with T to construct a feasible solution with objective value $Z^*_T(\mathbf{s})$ for problem (2.12); part (b) follows by using Theorem 2 and part (a); and part (c) follows by combining part (b) with Proposition 7. In Section 2.5, where we apply our fluid approach to multiarmed bandits, we provide numerical examples that show these three inequalities can be strict and the

bounds can be significantly different. The above result, Theorem 2, Proposition 4 and Proposition 7 can together be summarized in the following corollary.

Corollary 2 *When Assumptions 1 – 3 hold, for all* $\mathbf{s} \in \mathcal{S}$ *and all* $T \in \{1, 2, \dots\}$*:*

$$J^*(\mathbf{s}) \;\leq\; Z_T^*(\mathbf{s}) \;\leq\; \dots \;\leq\; Z_2^*(\mathbf{s}) \;\leq\; Z_1^*(\mathbf{s}) \;\leq\; Z_{ALO}^*(\mathbf{s}) \;=\; Z_{ALR}^*(\mathbf{s}) \;\leq\; Z_{CLR}^*(\mathbf{s}).$$

Theorem 3 essentially asserts that the fluid formulation provides a provably tighter bound than all three alternate approaches: the classical Lagrangian relaxation, the alternate Lagrangian relaxation and the ALO. The classical Lagrangian relaxation and the ALO formulations have been widely applied to solve practical problems; it is fair to say that these approaches constitute the state-of-the-art in solving large scale MDPs of practical interest. Our result is therefore significant because we have shown that the finite fluid formulation (2.3) leads to bounds that are at least as good as those of the Lagrangian relaxation and ALO formulations. In Section 2.5.4, we show that these difference can in fact be significant. More significantly, although Theorem 3 pertains to bounds, it is reasonable to expect that a formulation that produces a tighter bound will also produce better policies. Indeed, we will later show numerically that the heuristic policy given as Algorithm 1 based on the finite fluid problem (2.3) can significantly outperform the Lagrangian relaxation and ALO approaches.

2.4.5 Comparison of formulation sizes

As a complement to Theorem 3 where we compare the formulation bounds, we now compare the formulations in terms of their sizes. Table 2.1 summarizes the sizes of the four types of formulations in terms of the number of variables and the number of constraints. (Recall that q is the number of constraints that define the action space in equation (2.7) for the CLR approach.)

Although the exact numbers of variables and constraints will depend on the specific values of $|\mathcal{S}^m|$, $|\mathcal{A}|$, $|\mathcal{A}^m|$, T and q, we can derive some general qualitative insights:

Formulation	Number of variables	Number of constraints
ALO problem (2.5)	$\sum_{m=1}^{M} \lvert\mathcal{S}^m\rvert$	$\lvert\mathcal{S}\rvert \cdot \lvert\mathcal{A}\rvert$
CLR problem (2.8)	$\sum_{m=1}^{M} \lvert\mathcal{S}^m\rvert + q$	$\sum_{m=1}^{M} \lvert\mathcal{S}^m\rvert \cdot \lvert\mathcal{A}^m\rvert$
ALR problem (2.11)	$\sum_{m=1}^{M} \lvert\mathcal{S}^m\rvert + (M-1) \cdot \lvert\mathcal{A}\rvert$	$\sum_{m=1}^{M} \lvert\mathcal{S}^m\rvert \cdot \lvert\mathcal{A}\rvert$
Fluid problem (2.3)	$(T+1)\left(\sum_{m=1}^{M} \lvert\mathcal{S}^m\rvert \cdot \lvert\mathcal{A}\rvert + \lvert\mathcal{A}\rvert\right)$	$(T+1)\left(\sum_{m=1}^{M} \lvert\mathcal{S}^m\rvert + M\lvert\mathcal{A}\rvert\right)$

Table 2.1: Comparison of sizes of formulations. (The number of constraints quoted for each formulation does not count any nonnegativity constraints.)

- When Assumptions 1 – 3 hold and the action space can be described by a small number q of linking constraints as in equation (2.7), then the CLR problem (2.8) will in general be the smallest formulation, as its dimension are not dependent on $|\mathcal{A}|$ and $|\mathcal{S}|$.

- The largest formulation will in general be the ALO problem (2.5), as the number of constraints in the ALO scales with the size of the system state space $|\mathcal{S}|$ and the size of the system action space $|\mathcal{A}|$.

- The ALR problem (2.11) and the finite fluid formulation (2.3) will be somewhere in between the CLR problem (2.8) and the ALO problem (2.5), as the numbers of variables and constraints depend on the size of the system action space $|\mathcal{A}|$. Between the two, the fluid formulation will be larger than the ALR formulation due to the dependence on T.

Thus, while the fluid formulation provides a provably tighter bound than the other three formulations, it will in general not be the smallest formulation. In situations where the system action space $\mathcal{A}$ is not too large, the improved quality of the bound may justify the additional computational effort required for the fluid formulation.

2.4.6 Disaggregating the ALO and the ALR

The key idea in the fluid problem (2.3) is to partially disaggregate time in the first T periods, and then aggregate time in a discounted way from period $T+1$ on. This disaggregation is what allows us to prove that the fluid problem is tighter than the

ALR (part (a) of Theorem 3). One might then wonder if this type of disaggregation can be applied in the ALR and the ALO. To understand how this disaggregation applies in the ALR and ALO formulations, let us define two new partially disaggregated formulations: the ALR(T) formulation and the ALO(T) formulation.

The ALR(T) formulation is

$$\underset{\boldsymbol{\lambda},\mathbf{V}}{\text{minimize}} \quad \sum_{m=1}^{M} \sum_{k \in \mathcal{S}^m} \alpha_k^m(\mathbf{s}) V_k^m(1) \tag{2.13a}$$

$$\begin{aligned} \text{subject to} \quad V_k^m(t) \geq\ & g_{ka}^m - \mathbb{I}\{m < M\} \cdot \lambda_a^m(t) + \mathbb{I}\{m > 1\} \cdot \lambda_a^{m-1}(t) \\ & + \beta \cdot \sum_{j \in \mathcal{S}^m} p_{kja}^m V_j^m(t+1), \end{aligned}$$

$$\forall\ m \in \{1, \ldots, M\},\ k \in \mathcal{S}^m,\ a \in \mathcal{A},\ t \in \{1, \ldots, T\} \tag{2.13b}$$

$$\begin{aligned} V_k^m(T+1) \geq\ & g_{ka}^m - \mathbb{I}\{m < M\} \cdot \lambda_a^m(T+1) + \mathbb{I}\{m > 1\} \cdot \lambda_a^{m-1}(T+1) \\ & + \beta \cdot \sum_{j \in \mathcal{S}^m} p_{kja}^m V_j^m(T+1), \end{aligned}$$

$$\forall\ m \in \{1, \ldots, M\},\ k \in \mathcal{S}^m,\ a \in \mathcal{A}. \tag{2.13c}$$

Let $Z^*_{ALR(T)}(\mathbf{s})$ denote the optimal value of problem (2.13).

The ALO(T) formulation is

$$\underset{\mathbf{J}}{\text{minimize}} \quad \sum_{m=1}^{M} \sum_{k \in \mathcal{S}^m} \alpha_k^m(\mathbf{s}) \cdot J_k^m(1) \tag{2.14a}$$

$$\begin{aligned} \text{subject to} \quad & \sum_{m=1}^{M} J_{\bar{s}^m}^m(t) \geq \sum_{m=1}^{M} g_{\bar{s}^m a}^m + \beta \sum_{m=1}^{M} \sum_{j \in \mathcal{S}^m} p_{\bar{s}^m ja}^m J_j^m(t+1), \\ & \forall\ \bar{\mathbf{s}} \in \mathcal{S},\ a \in \mathcal{A},\ t \in \{1, \ldots, T\}, \end{aligned} \tag{2.14b}$$

$$\sum_{m=1}^{M} J_{\bar{s}^m}^m(T+1) \geq \sum_{m=1}^{M} g_{\bar{s}^m a}^m + \beta \sum_{m=1}^{M} \sum_{j \in \mathcal{S}^m} p_{\bar{s}^m ja}^m J_j^m(T+1), \quad \forall\ \bar{\mathbf{s}} \in \mathcal{S},\ a \in \mathcal{A}. \tag{2.14c}$$

Let $Z^*_{ALO(T)}(\mathbf{s})$ denote the optimal value of problem (2.14).

We then have the following theoretical result.

Theorem 4 *For all* $\mathbf{s} \in \mathcal{S}$, $T \in \{1, 2, \dots\}$, *we have* $Z_T^*(\mathbf{s}) = Z_{ALR(T)}^*(\mathbf{s}) = Z_{ALO(T)}^*(\mathbf{s})$.

The first part of the equality re-states more rigorously the earlier observation from Section 2.4.3, which is that the fluid problem can be viewed as the ALR problem with time disaggregated over a horizon of T periods. The second equality asserts that the fluid problem is equivalent to a time-disaggregated version of the ALO, analogously to Theorem 2.

2.5 Application to multiarmed bandit problems

2.5.1 Problem definition

In the multiarmed bandit problem, the decision maker is presented with a set of bandits/arms, and each arm is endowed with some state space. At each point in time, the decision maker needs to select one of the arms to activate so as to maximize his long-term (over an infinite horizon) expected discounted reward. We consider the regular multiarmed bandit problem, where only the activated arm changes state and generates reward, and the restless multiarmed bandit problem, where inactive arms may also change state state (i.e., passive transitions are allowed) and generate reward (i.e., there are passive rewards).

2.5.2 Fluid model

The multiarmed problem can be readily formulated in our fluid framework. The components M of the stochastic system correspond to the individual bandits. The action space $\mathcal{A}$ is defined here as $\mathcal{A} = \{1, \dots, M\}$. The reward g_{ka}^m, for $m \in \{1, \dots, M\}$, $k \in \mathcal{S}^m$ and $a \in \mathcal{A}$ is the reward that is earned when arm m is in state k and arm a is activated. Similarly, p_{kja}^m is the probability that bandit m transitions from state $k \in \mathcal{S}^m$ to state $j \in \mathcal{S}^m$ when arm a is activated. In the case of the regular bandit problem, we need to ensure that whenever $m \neq a$, $g_{ka}^m = 0$ for every $k \in \mathcal{S}^m$ and $p_{kja}^m = \mathbb{I}\{k = j\}$ for every pair of states $k, j \in \mathcal{S}^m$. In the case of the restless bandit

problem, we only need to ensure that whenever $a, a', m \in \{1, \ldots, M\}$, with $m \neq a$ and $m \neq a'$, that $g^m_{ka} = g^m_{ka'}$ for every $k \in \mathcal{S}^m$ and $p^m_{kja} = p^m_{kja'}$ for every $k, j \in \mathcal{S}^m$.

2.5.3 Relation to [19]

One interesting property of the fluid formulation is how it relates to the performance measure formulation developed in [19]. Let w^m_{j0} be defined for every bandit $m \in \{1, \ldots, M\}$, state $j \in \mathcal{S}^m$ as the expected discounted number of times that bandit m is in state j and it is not activated. Similarly, let w^m_{j1} be defined as the expected discounted number of times that bandit m is in state j and *is* activated. Let $\bar{p}^m_{ij1}$ and $\bar{p}^m_{ij0}$ be the active and passive transition probabilities of bandit m from state i to state j respectively, and let R^m_{k0} and R^m_{k1} be the passive and active rewards from activating bandit m when it is in state k, respectively. The w^m_{ja} variables are referred to as *performance measures*. Finally, suppose that the system starts in state $\mathbf{s} \in \mathcal{S}$ at time $t = 1$ and as assumed in Section 2.5.1, we must activate exactly one arm at any period.

For a given collection of performance measures, the corresponding reward is the sum $\sum_{m=1}^{M} \sum_{k \in \mathcal{S}^m} \sum_{a \in \{0,1\}} R^m_{ka} w^m_{ka}$, which forms the objective of the problem. The performance measures, by their definition, satisfy certain conservation laws, and the feasible set of performance measures resulting from those laws is referred to as the *performance region*. The formulation developed in [19] is to maximize this reward over the performance region:

$$\underset{\mathbf{w}}{\text{maximize}} \quad \sum_{m=1}^{M} \sum_{k \in \mathcal{S}^m} \sum_{a \in \{0,1\}} R^m_{ka} w^m_{ka} \tag{2.15a}$$

$$\text{subject to} \quad \sum_{m=1}^{M} \sum_{k \in \mathcal{S}^m} w^m_{k1} = \frac{1}{1-\beta}, \tag{2.15b}$$

$$w^m_{j0} + w^m_{j1} = \alpha^m_j(\mathbf{s}) + \beta \sum_{i \in \mathcal{S}^m} \sum_{a \in \{0,1\}} \bar{p}^m_{ija} w^m_{ia}, \quad \forall\, m \in \{1, \ldots, M\},\ j \in \mathcal{S}^m, \tag{2.15c}$$

$$w^m_{j0}, w^m_{j1} \geq 0, \quad \forall\, m \in \{1, \ldots, M\},\ j \in \mathcal{S}^m. \tag{2.15d}$$

In words, the formulation finds the vector of performance measures $\mathbf{w}$ that satisfies the transition constraints at the level of the components and maximizes the total expected discounted reward, which is just the sum of the performance measures weighted by their corresponding rewards. Note that in terms of the data defining the fluid formulation, the data in problem (2.15) and in the finite fluid problem (2.3) identify as follows. For every bandit m, states i and j, we have $\bar{p}^m_{ij1} = p^m_{ijm}$, while $\bar{p}^m_{ij0} = p^m_{ija}$ for every $a \neq m$. Similarly, for the rewards, we have $R^m_{k1} = g^m_{km}$ for every bandit m and state k, while $R^m_{k0} = g^m_{ka}$ for every $a \neq m$. To make decisions, [19] propose a primal dual heuristic where one solve problem (2.15) at each new state $\mathbf{s}$. We describe how the heuristic operates for the case when exactly one arm must be activated at each period; for more details, the interested reader is referred to [19]. Using the solution of the problem, the heuristic considers the optimal variables $w^m_{s^m 1}$ and $w^m_{s^m 0}$ for each $m \in \{1, \ldots, M\}$ and proceeds as follows:

1. If exactly one of $w^m_{s^m 1}$ for $m \in \{1, \ldots, M\}$ is positive, say bandit m', then activate bandit m'. (Intuitively, $w^m_{s^m 1}$ represents the expected discounted amount of time that bandit m is in its initial state s^m and it is activated; if there is only one bandit for which this value is positive, the solution suggests that we should activate this bandit.)

2. If all $w^m_{s^m 1}$ are zero, then activate the bandit $m \in \{1, \ldots, M\}$ with the lowest reduced cost of the active performance measure $w^m_{s^m 1}$. (Intuitively, the reduced cost of $w^m_{s^m 1}$ represents the marginal decrease in the objective value per unit increase in $w^m_{s^m 1}$; by selecting the m with the lowest reduced cost of $w^m_{s^m 1}$ we select the bandit that will have the lowest detriment to the objective.)

3. If more than one $w^m_{s^m 1}$ for $m \in \{1, \ldots, M\}$ is positive, then activate the bandit m with the largest reduced cost of the passive performance measure $w^m_{s^m 0}$ among those m with $w^m_{s^m 1} > 0$. (Similarly to the previous case, the reduced cost of $w^m_{s^m 0}$ represents the marginal decrease in the objective for a unit increase in the passive performance measure $w^m_{s^m 0}$; by activating the bandit with the largest passive reduced cost we try to counteract this effect.)

Before continuing on to the results, it is important to establish how the performance region formulation relates to the formulations presented in Section 2.4 and to the fluid method. Let $Z^*_{BNM}(\mathbf{s})$ be the optimal objective value of problem (2.15) when the system starts in state $\mathbf{s} \in \mathcal{S}$. First, problem (2.15) and the CLR problem (2.8) are equivalent; this connection was originally observed by [58]. To see this, for each m set $\mathcal{A}^m = \{0, 1\}$, where 1 indicates that bandit m is activated and 0 indicates that it is not activated, and plug in the following choices of $\mathbf{D}^m(\cdot)$ and $\mathbf{b}$ for the coupling constraint in equation (2.7):

$$\mathbf{D}^m(a^m) = \begin{bmatrix} \mathbb{I}\{a^m = 1\} \\ -\mathbb{I}\{a^m = 1\} \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 1 \\ -1 \end{bmatrix}. \tag{2.16}$$

This coupling constraint requires that exactly one bandit be activated. It is then easy to see that the dual CLR problem (2.9) exactly coincides with problem (2.15), leading to the following result.

Proposition 8 *For each* $\mathbf{s} \in \mathcal{S}$, $Z^*_{CLR}(\mathbf{s}) = Z^*_{BNM}(\mathbf{s})$.

It turns out that problem (2.15) and the ALR problem (2.11) for the problem as defined in Section 2.5.1 are in fact the same, in that they lead to the same objective value.

Proposition 9 *For each* $\mathbf{s} \in \mathcal{S}$, $Z^*_{ALR}(\mathbf{s}) = Z^*_{BNM}(\mathbf{s})$.

The proof (found in Section A.1.9 of Appendix A) consists of showing that the optimal solution of one can be used to construct a feasible solution for the other. An immediate corollary of this result and Theorem 3 is that the finite fluid formulation bound $Z^*_T(\mathbf{s})$ is at least as tight as the performance region bound $Z^*_{BNM}(\mathbf{s})$.ă

Corollary 3 *For each* $\mathbf{s} \in \mathcal{S}$ *and* $T \in \{1, 2, \dots\}$, $Z^*_T(\mathbf{s}) \leq Z^*_{BNM}(\mathbf{s})$.

Combining Theorem 2 and Propositions 4, 8 and 9, we obtain the following corollary which summarizes the ordering of all bounds for this problem.

Corollary 4 *For the bandit problem defined in Section 2.5.1, for all* $\mathbf{s} \in \mathcal{S}$ *and* $T \in \{1, 2, \dots\}$*:*

$$J^*(\mathbf{s}) \leq Z_T^*(\mathbf{s}) \leq \cdots \leq Z_2^*(\mathbf{s}) \leq \ Z_1^*(\mathbf{s}) \leq Z_{ALO}^*(\mathbf{s}) = Z_{ALR}^*(\mathbf{s}) = Z_{CLR}^*(\mathbf{s}) = Z_{BNM}^*(\mathbf{s}).$$

In Sections 2.5.4, we will show that the inequality between $Z_T^*(\mathbf{s})$ and the four equivalent bounds – $Z_{ALO}^*(\mathbf{s})$, $Z_{ALR}^*(\mathbf{s})$, $Z_{CLR}^*(\mathbf{s})$ and $Z_{BNM}^*(\mathbf{s})$ – can be strict.

Since the fluid formulation provides a bound that is at least as tight as the performance region formulation, it would seem reasonable to expect that the heuristic policy derived from the fluid formulation to give performance that is generally as good as, if not better than, that of the primal dual heuristic derived from the performance region formulation in [19]. In Section 2.5.5, we will show that this is indeed the case, and that in fact the fluid-based heuristic significantly outperforms the primal dual heuristic of [19].

2.5.4 Bound comparison

We begin the discussion of our numerical results by comparing the bound generated by our fluid optimization model to the bound generated by the ALR on medium-scale instances. For the fluid approach, we considered T values of 1, 5 and 10; for values of $T > 10$, the metric values changed negligibly relative to $T = 10$.

We set the number of bandits M to 5 and the number of states of each bandit n to 4, resulting in $5^4 = 1024$ system states. In each bandit state space, we number the states from 1 to n, i.e., $\mathcal{S}^m = \{1, 2, \dots, n\}$. We generated four different sets of five instances, with the following structure:

- REG.SAR, consisting of regular multiarmed bandits, where the reward g_{km}^m was set as $g_{km}^m = (10/n) \cdot k$ for every bandit m and state k. Each active transition probability vector was drawn uniformly from the $(n-1)$-dimensional unit simplex.

- RSTLS.SAR, consisting of restless bandits, with the same reward structure as

REG.SAR. Each active and passive transition probability vector was drawn uniformly from the $(n-1)$-dimensional unit simplex.

- RSTLS.SBR, consisting of restless bandits, where the active reward g_{km}^m was set as $g_{km}^m = (10/n) \cdot k$ for every m and k, and the passive reward g_{ka}^m for $a \neq m$ was set to ρ_k^m, where $\rho_k^m = (1/M) \cdot (10/n) \cdot k$ for each m and k. Each active and passive transition probability vector was drawn uniformly from the $(n-1)$-dimensional unit simplex.

- RSTLS.DET.SBR, consisting of restless bandits, where the reward structure is the same as RSTLS.SBR. Each active and passive transition probability matrix was generated by permuting the rows of the n-dimensional identity matrix uniformly at random (i.e., transition matrices are still randomly generated, but the transitions that they govern are now *deterministic*).

The reason for considering the types of reward structures in sets REG.SAR, RSTLS.SAR, RSTLS.SBR and RSTLS.DET.SBR is that in these sets of instances, the reward structures of any two bandits are identical, but they are different in their probabilistic structure. In order for a method to be successful, therefore, it must be able to recognize that the bandits are different in their probabilistic structure, which will directly affect the long-term expected reward that the method could possibly garner from each bandit. We would expect that the greedy method, which only uses reward information, would perform rather poorly on these instances. RSTLS.SAR and RSTLS.SBR are interesting to consider together because passive rewards are zero in the former and non-zero in the latter. RSTLS.DET.SBR is interesting as it is not stochastic and thus constitutes a potentially pathological instance set.

To compare the bounds, we define three different metrics as follows. Given a method h for solving the problem that is based on an optimization formulation, let $Z_h(\mathbf{s})$ be the objective value (upper bound) generated at $\mathbf{s}$. Define $\mathrm{RD}_h(\mathbf{s}) = 100\% \times (Z_h(\mathbf{s}) - J^*(\mathbf{s}))/J^*(\mathbf{s})$ as the relative difference between $Z_h(\mathbf{s})$ and the optimal value function $J^*(\mathbf{s})$. Then, define the metrics $\mathcal{O}_{\mathrm{mean},h}$, $\mathcal{O}_{P,h}$ and $\mathcal{O}_{\mathrm{max},h}$ as the mean, Pth percentile and maximum of $\{\mathrm{RD}_h(\mathbf{s})\}_{\mathbf{s}\in\mathcal{S}}$. In general, the lower the values of

the $\mathcal{O}$ metrics, the closer the bound is to the true optimal value function; a value of zero for $\mathcal{O}_{\text{mean},h}$ implies that the bound/objective value is exactly equal to the true optimal value function. We will consider these metrics for the fluid and the ALR formulations. We compute the optimal value function J^* using value iteration.

Tables 2.2 and 2.3 compare the objective values obtained from the fluid formulation and from the ALR problem (2.11) with the optimal objective value. We only show the first instance from each set, as the results for the other instances in each instance set were qualitatively similar. Recall that by Theorem 2, Proposition 8 and Proposition 9 that the ALR problem (2.11), the CLR problem (2.8), the ALO problem (2.5) and the performance region problem (2.15) all yield the same objective value for a fixed initial state $\mathbf{s}$. We can see that for every instance and every discount factor, the objective values from the fluid formulation are closer to the optimal objective than those from the ALR and by extension, those from the ALO, CLR and performance region formulations. We can also see that although part (b) of Theorem 3 indicates that the fluid bound does not worsen as T increases, there is negligible improvement beyond the $T = 5$ to $T = 10$ range. This suggests that we can obtain substantially better state-wise bounds than the ALR formulation by solving only a modestly larger LO problem.

2.5.5 Large scale bandit results

So far, we have focused on bandit problems that are relatively small and have compared the bounds from the different methods. In this section, we compare the *policy performance* of the methods on larger instances, where the optimal value function is unavailable to us and where we must resort to simulation. We consider instances that are generated in the same way as the RSTLS.DET.SBR set of the previous section, for values of M in $\{5, 10, 15, 20\}$ and values of n in $\{5, 10, 20\}$. We restrict ourselves to a discount factor of $\beta = 0.99$ and simulate the system for 500 steps. We consider the fluid heuristic with T values of 1, 2, 5 and 10, the ALR approach, the primal dual heuristic of [19] and the greedy heuristic (which activates the arm that leads to the highest immediate reward). For the ALR approach, we re-solve it at each new

Set	Instance	β	Method (h)	$\mathcal{O}_{\text{mean},h}$	$\mathcal{O}_{95,h}$	$\mathcal{O}_{\max,h}$
REG.SAR	1	0.5	Fluid, $T = 1$	1.3511	2.7801	4.3253
			Fluid, $T = 5$	0.4161	0.8019	**1.2562**
			Fluid, $T = 10$	**0.4015**	**0.7976**	**1.2562**
			ALR	2.0973	5.2572	7.0816
		0.9	Fluid, $T = 1$	1.8020	3.9777	6.7438
			Fluid, $T = 5$	0.6383	1.2568	1.9959
			Fluid, $T = 10$	**0.3879**	**0.6368**	**0.6997**
			ALR	2.5094	5.4317	8.7091
		0.95	Fluid, $T = 1$	0.7311	1.7283	2.7162
			Fluid, $T = 5$	0.2679	0.6317	0.9857
			Fluid, $T = 10$	**0.1518**	**0.2938**	**0.3639**
			ALR	0.9774	2.2040	3.4264
		0.99	Fluid, $T = 1$	0.0334	0.0862	0.1391
			Fluid, $T = 5$	0.0132	0.0343	0.0556
			Fluid, $T = 10$	**0.0081**	**0.0170**	**0.0224**
			ALR	0.0436	0.1063	0.1708
RSTLS.SAR	1	0.5	Fluid, $T = 1$	2.9222	5.2561	6.8915
			Fluid, $T = 5$	**2.6406**	**4.1177**	**5.2122**
			Fluid, $T = 10$	**2.6406**	**4.1177**	**5.2122**
			ALR	4.5558	10.0487	52.7464
		0.9	Fluid, $T = 1$	4.7079	5.6050	6.5985
			Fluid, $T = 5$	**4.5995**	**5.1573**	**5.5452**
			Fluid, $T = 10$	**4.5995**	**5.1573**	**5.5452**
			ALR	5.4381	8.3128	13.6692
		0.95	Fluid, $T = 1$	4.8905	5.3635	5.8881
			Fluid, $T = 5$	**4.8336**	**5.1322**	**5.3391**
			Fluid, $T = 10$	**4.8336**	**5.1322**	**5.3391**
			ALR	5.2499	6.6973	9.2057
		0.99	Fluid, $T = 1$	5.0305	5.1297	5.2383
			Fluid, $T = 5$	**5.0187**	**5.0810**	**5.1239**
			Fluid, $T = 10$	**5.0187**	**5.0810**	**5.1239**
			ALR	5.1015	5.3931	5.8688

Table 2.2: Objective value results (in %) for infinite horizon experiment, $M = 5$, $n = 4$, for instance 1 of sets REG.SAR and RSTLS.SAR. In each instance, value of β and metric, the best value is indicated in bold.

Set	Instance	β	Method (h)	$\mathcal{O}_{\text{mean},h}$	$\mathcal{O}_{95,h}$	$\mathcal{O}_{\max,h}$
RSTLS.SBR	1	0.5	Fluid, $T = 1$	1.1795	2.0612	3.2705
			Fluid, $T = 5$	**1.1426**	**1.8816**	**2.4596**
			Fluid, $T = 10$	**1.1426**	**1.8816**	**2.4596**
			ALR	2.5059	8.1574	27.8741
		0.9	Fluid, $T = 1$	2.1639	2.5757	2.7914
			Fluid, $T = 5$	**2.1286**	**2.4244**	**2.6105**
			Fluid, $T = 10$	**2.1286**	**2.4244**	**2.6105**
			ALR	2.6231	4.3336	6.5445
		0.95	Fluid, $T = 1$	2.2758	2.4911	2.6111
			Fluid, $T = 5$	**2.2564**	**2.4104**	**2.5125**
			Fluid, $T = 10$	**2.2564**	**2.4104**	**2.5125**
			ALR	2.5039	3.3414	4.3872
		0.99	Fluid, $T = 1$	2.3645	2.4090	2.4350
			Fluid, $T = 5$	2.3604	2.3927	2.4140
			Fluid, $T = 10$	**2.3604**	**2.3927**	**2.4140**
			ALR	2.4099	2.5735	2.7749
RSTLS.DET.SBR	1	0.5	Fluid, $T = 1$	0.5910	3.2775	7.7156
			Fluid, $T = 5$	0.0403	0.2744	0.6560
			Fluid, $T = 10$	**0.0152**	**0.0664**	**0.6012**
			ALR	1.7255	7.8925	23.0435
		0.9	Fluid, $T = 1$	0.6112	2.2488	3.5360
			Fluid, $T = 5$	0.0975	0.4078	**1.4104**
			Fluid, $T = 10$	**0.0816**	**0.3973**	**1.4104**
			ALR	1.0275	3.3645	4.6934
		0.95	Fluid, $T = 1$	0.4020	1.3776	2.1960
			Fluid, $T = 5$	0.0662	0.3673	**0.9590**
			Fluid, $T = 10$	**0.0609**	**0.3446**	**0.9590**
			ALR	0.5910	1.8808	2.6698
		0.99	Fluid, $T = 1$	0.1102	0.3352	0.5066
			Fluid, $T = 5$	0.0223	0.0995	**0.2475**
			Fluid, $T = 10$	**0.0158**	**0.0984**	**0.2475**
			ALR	0.1497	0.3972	0.5404

Table 2.3: Objective value results (in %) for infinite horizon experiment, $M = 5$, $n = 4$, for instance 1 of sets RSTLS.SBR and RSTLS.DET.SBR. In each instance, value of β and metric, the best value is indicated in bold.

state $\mathbf{s}$, and take the action that is greedy with respect to the value function approximation $\mathbf{V}$. Note that we do not consider the policy that is greedy with respect to the value function approximation from the ALO formulation (2.5) since by part (b) of Theorem 2, any value function approximation that is optimal for the ALR (2.11) is a value function approximation that is optimal for the ALO (2.5), and vice versa. Similarly, we do not consider the policy that arises from the CLR formulation (2.8), since by Propositions 8 and 8, the CLR and ALR formulations are equivalent for this problem.

To compare the methods, for each pair (M, n), we generate $K = 100$ random initial states $\mathbf{s}^{(1)}, \ldots, \mathbf{s}^{(K)}$ by uniformly selecting one of the n states for each component. We simulate each policy h from each initial state $\mathbf{s}^{(k)}$ to obtain a realized reward $J_{k,h}$. We also compute the initial objective value $Z_{k,h}$ of the policy h (where applicable) at each initial state. For each initial state $\mathbf{s}^{(k)}$ and method h, we thus obtain a gap value $G_{k,h}$, defined as

$$G_{k,h} = 100\% \times \frac{Z_k^* - J_{k,h}}{Z_k^*}$$

where $Z_k^* = \min_h Z_{k,h}$ is the lowest upper bound available (in this set of experiments, this is the fluid method with the largest value of T). We then consider the mean value of $\{G_{k,h}\}_{k=1}^K$ for each method h, which we report as $G_{\text{mean},h}$. In addition, for each initial state $\mathbf{s}^{(k)}$ and method h based on a mathematical optimization formulation, we compute the relative difference $U_{k,h}$ between the upper bound from h and the best upper bound, defined as

$$U_{k,h} = 100\% \times \frac{Z_{k,h} - Z_k^*}{Z_k^*},$$

and we compute the mean over the K initial states as $U_{\text{mean},h}$. Finally, for each initial state $\mathbf{s}^{(k)}$ and each method h that is based on an optimization formulation, we compute $T_{k,h}$, which is the average solution time in seconds of the underlying formulation over all of the steps of the simulation. We then consider the mean value of $\{T_{k,h}\}_{k=1}^K$ for each applicable method h, which we report as $T_{\text{mean},h}$.

Tables 2.4 and 2.5 display the results from this collection of instances. With regard to policy performance, the results indicate that the fluid method delivers excellent

performance, even in the most challenging instance ($M = 20$, $n = 20$), and significantly outperforms the greedy heuristic, the Lagrangian relaxation approach and the primal dual heuristic. From a solution time perspective, the finite fluid formulation (2.3) does take considerably more time per action than either the performance region formulation (2.15) or the ALR formulation (2.11). However, even in the largest case ($M = 20$, $n = 20$) and for the largest value of T, the average time per action is on the order of 2.6 seconds; for certain applications, this amount of time may still be feasible.

2.6 Conclusion

In this chapter, we have considered a fluid optimization approach for solving decomposable MDPs. The essential feature of the approach is that it models the transitions of the system at the level of individual components; in this way, the approach is tractable and scalable. We provided theoretical justification for this approach by showing that it provides tighter bounds on the optimal value than three state-of-the-art approaches. We showed computationally that this approach leads to strong performance in multiarmed bandit problems.

There are several promising directions for future research. It would be valuable to extend the approach to deal with situations where the data (e.g., the transition probabilities) are not known precisely and may become known more precisely with time. Problems of this kind fall in the domain of robust optimization (see [15]) and it would seem that an adaptable robust version of the fluid formulation could be appropriate in this setting. At the same time, problems of this kind could also be viewed as reinforcement learning problems. One approach from this direction could involve combining the fluid approach with posterior sampling (see, e.g., [83]): in this approach, one would maintain a distribution over the problem data and at each period, one would take a sample from this distribution, solve the fluid problem corresponding to the sample to determine the action to take and update the distribution with the realized reward and transitions from that action. Exploring the benefits of such a

Instance	β	Method (h)	$G_{\text{mean},h}$	SE	$U_{\text{mean},h}$	SE	$T_{\text{mean},h}$	SE
$M = 5$, $n = 5$	0.99	Fluid, $T = 1$	10.3367	(0.352)	0.2319	(0.009)	0.002	(0.00)
		Fluid, $T = 2$	6.4168	(0.016)	0.1870	(0.008)	0.002	(0.00)
		Fluid, $T = 5$	4.0224	(0.008)	0.0976	(0.006)	0.006	(0.00)
		Fluid, $T = 10$	4.2265	(0.062)	0.0000	(0.000)	0.016	(0.00)
		Greedy	24.9822	(0.255)	–	–	–	–
		ALR	9.2136	(0.079)	0.2858	(0.009)	0.003	(0.00)
		BNMPD	31.6512	(1.407)	0.2858	(0.009)	0.001	(0.00)
$M = 5$, $n = 10$	0.99	Fluid, $T = 1$	4.6314	(0.117)	0.1898	(0.008)	0.003	(0.00)
		Fluid, $T = 2$	4.4024	(0.120)	0.1530	(0.007)	0.003	(0.00)
		Fluid, $T = 5$	2.7667	(0.039)	0.0664	(0.004)	0.009	(0.00)
		Fluid, $T = 10$	2.7258	(0.032)	0.0000	(0.000)	0.033	(0.00)
		Greedy	21.1001	(0.395)	–	–	–	–
		ALR	13.0486	(0.377)	0.2389	(0.010)	0.007	(0.00)
		BNMPD	31.9437	(0.690)	0.2389	(0.010)	0.002	(0.00)
$M = 5$, $n = 20$	0.99	Fluid, $T = 1$	7.4533	(0.190)	0.2898	(0.011)	0.005	(0.00)
		Fluid, $T = 2$	6.7666	(0.079)	0.2254	(0.009)	0.005	(0.00)
		Fluid, $T = 5$	5.8754	(0.171)	0.1166	(0.006)	0.014	(0.00)
		Fluid, $T = 10$	5.2519	(0.192)	0.0000	(0.000)	0.056	(0.00)
		Greedy	22.2220	(0.334)	–	–	–	–
		ALR	18.3783	(0.194)	0.3557	(0.012)	0.013	(0.00)
		BNMPD	36.8163	(0.724)	0.3557	(0.012)	0.004	(0.00)
$M = 10$, $n = 5$	0.99	Fluid, $T = 1$	3.4065	(0.152)	0.0860	(0.005)	0.006	(0.00)
		Fluid, $T = 2$	1.6663	(0.147)	0.0596	(0.004)	0.008	(0.00)
		Fluid, $T = 5$	1.0393	(0.068)	0.0209	(0.002)	0.032	(0.00)
		Fluid, $T = 10$	1.0984	(0.085)	0.0000	(0.000)	0.091	(0.00)
		Greedy	10.2405	(0.406)	–	–	–	–
		ALR	8.4630	(0.578)	0.1218	(0.007)	0.008	(0.00)
		BNMPD	34.9582	(1.601)	0.1218	(0.007)	0.002	(0.00)
$M = 10$, $n = 10$	0.99	Fluid, $T = 1$	1.8662	(0.117)	0.1611	(0.007)	0.010	(0.00)
		Fluid, $T = 2$	1.4467	(0.066)	0.1263	(0.006)	0.013	(0.00)
		Fluid, $T = 5$	1.4348	(0.095)	0.0532	(0.003)	0.056	(0.00)
		Fluid, $T = 10$	1.4386	(0.107)	0.0000	(0.000)	0.150	(0.00)
		Greedy	18.4851	(0.334)	–	–	–	–
		ALR	16.0600	(0.873)	0.1902	(0.007)	0.022	(0.00)
		BNMPD	34.3912	(1.470)	0.1902	(0.007)	0.003	(0.00)
$M = 10$, $n = 20$	0.99	Fluid, $T = 1$	2.8752	(0.130)	0.3040	(0.010)	0.018	(0.00)
		Fluid, $T = 2$	2.5045	(0.090)	0.2722	(0.011)	0.023	(0.00)
		Fluid, $T = 5$	2.1928	(0.060)	0.1579	(0.009)	0.088	(0.00)
		Fluid, $T = 10$	1.8246	(0.061)	0.0000	(0.000)	0.313	(0.00)
		Greedy	22.6095	(0.223)	–	–	–	–
		ALR	26.9484	(0.416)	0.3488	(0.011)	0.068	(0.00)
		BNMPD	40.5795	(0.381)	0.3488	(0.011)	0.004	(0.00)

Table 2.4: Large scale policy performance and runtime simulation results for $M \in \{5, 10\}$, $n \in \{5, 10, 20\}$ RSTLS.DET.SBR instances. (SE indicates standard error.)

Instance	β	Method (h)	$G_{\text{mean},h}$	SE	$U_{\text{mean},h}$	SE	$T_{\text{mean},h}$	SE
$M = 15$, $n = 5$	0.99	Fluid, $T = 1$	0.7693	(0.030)	0.0359	(0.003)	0.020	(0.00)
		Fluid, $T = 2$	0.7681	(0.032)	0.0172	(0.002)	0.028	(0.00)
		Fluid, $T = 5$	0.7015	(0.013)	0.0039	(0.001)	0.143	(0.00)
		Fluid, $T = 10$	0.7001	(0.013)	0.0000	(0.000)	0.218	(0.00)
		Greedy	9.1309	(0.216)	–	–	–	–
		ALR	4.3748	(0.428)	0.0523	(0.004)	0.022	(0.00)
		BNMPD	28.2468	(0.800)	0.0523	(0.004)	0.002	(0.00)
$M = 15$, $n = 10$	0.99	Fluid, $T = 1$	2.2354	(0.054)	0.1176	(0.006)	0.025	(0.00)
		Fluid, $T = 2$	1.5200	(0.023)	0.0931	(0.005)	0.036	(0.00)
		Fluid, $T = 5$	1.3230	(0.006)	0.0355	(0.003)	0.137	(0.00)
		Fluid, $T = 10$	1.1694	(0.004)	0.0000	(0.000)	0.533	(0.00)
		Greedy	11.8774	(0.160)	–	–	–	–
		ALR	21.4492	(0.302)	0.1443	(0.006)	0.091	(0.00)
		BNMPD	34.4392	(0.535)	0.1443	(0.006)	0.004	(0.00)
$M = 15$, $n = 20$	0.99	Fluid, $T = 1$	2.1894	(0.081)	0.2589	(0.010)	0.046	(0.00)
		Fluid, $T = 2$	2.0305	(0.040)	0.2312	(0.011)	0.062	(0.00)
		Fluid, $T = 5$	1.7992	(0.032)	0.1106	(0.006)	0.217	(0.00)
		Fluid, $T = 10$	1.6754	(0.036)	0.0000	(0.000)	1.170	(0.00)
		Greedy	13.2699	(0.106)	–	–	–	–
		ALR	14.6558	(0.174)	0.2893	(0.011)	0.354	(0.00)
		BNMPD	37.7471	(0.626)	0.2893	(0.011)	0.005	(0.00)
$M = 20$, $n = 5$	0.99	Fluid, $T = 1$	1.1169	(0.045)	0.0554	(0.004)	0.056	(0.00)
		Fluid, $T = 2$	0.8508	(0.008)	0.0318	(0.003)	0.058	(0.00)
		Fluid, $T = 5$	0.7976	(0.004)	0.0065	(0.001)	0.148	(0.00)
		Fluid, $T = 10$	0.7953	(0.004)	0.0000	(0.000)	0.418	(0.00)
		Greedy	9.9393	(0.228)	–	–	–	–
		ALR	8.2415	(0.916)	0.0816	(0.005)	0.043	(0.00)
		BNMPD	30.1071	(0.724)	0.0816	(0.005)	0.003	(0.00)
$M = 20$, $n = 10$	0.99	Fluid, $T = 1$	2.6674	(0.039)	0.1360	(0.005)	0.076	(0.00)
		Fluid, $T = 2$	2.6459	(0.032)	0.1104	(0.005)	0.069	(0.00)
		Fluid, $T = 5$	1.5767	(0.025)	0.0416	(0.003)	0.278	(0.00)
		Fluid, $T = 10$	1.3553	(0.024)	0.0000	(0.000)	1.206	(0.01)
		Greedy	9.7231	(0.106)	–	–	–	–
		ALR	12.4764	(0.206)	0.1632	(0.006)	0.203	(0.00)
		BNMPD	33.3620	(0.786)	0.1632	(0.006)	0.004	(0.00)
$M = 20$, $n = 20$	0.99	Fluid, $T = 1$	5.1385	(0.105)	0.1918	(0.007)	0.086	(0.00)
		Fluid, $T = 2$	3.3105	(0.085)	0.1614	(0.006)	0.100	(0.00)
		Fluid, $T = 5$	1.5725	(0.039)	0.0822	(0.004)	0.487	(0.00)
		Fluid, $T = 10$	1.2369	(0.026)	0.0000	(0.000)	2.657	(0.01)
		Greedy	11.2028	(0.134)	–	–	–	–
		ALR	11.9748	(0.090)	0.2238	(0.007)	0.089	(0.00)
		BNMPD	36.2378	(0.759)	0.2238	(0.007)	0.005	(0.00)

Table 2.5: Large scale policy performance and runtime simulation results for $\beta = 0.99$, $M \in \{15, 20\}$, $n \in \{5, 10, 20\}$ RSTLS.DET.SBR instances. (SE indicates standard error.)

scheme, as well as other ways of combining the fluid method with reinforcement learning methods, thus constitutes another interesting direction of future work.

Chapter 3

Robust product line design

3.1 Introduction

A *product line* is a collection of products that are variations of a single basic product, differing with respect to certain attributes. Firms offer product lines to account for heterogeneity in consumer preferences. For example, consider a producer of breakfast cereals. The basic product may be a type of cereal (e.g., corn flakes), and a product line may consist of versions of this cereal that differ with respect to attributes such as the size of the cereal box, the flavor, whether the cereal has low fat content, whether it has low calorie content, whether it is nutritionally enriched with certain vitamins, what its retail price is, and so on. The customers of this cereal company may include: weight-conscious adults, who may prefer healthy versions of the cereal; college students, who may prefer certain flavors and larger packages (with lower cost per unit weight); or fitness-oriented adults, who may prefer enriched cereal that has added vitamins and minerals in smaller packages and who may be willing to pay more than other customers. In this case, it is clear that there is no single cereal product that would be highly desirable for all three groups. Rather, it would be more appropriate to introduce several versions of the cereal that, taken together, would appeal to all three categories of customers.

The problem of *product line design* (PLD) is to select the attributes of the products comprising the product line so as to maximize the revenue that will result from the

different types of preferences within the customer population. This problem is one of vital importance to the success of a firm. Much research effort has been devoted to both modeling and solving the PLD problem.

The key prerequisite to practically all existing PLD approaches is a choice model, which specifies the probability that a random customer selects one of the products in the product line or opts to not purchase any of them, given the set of products that is offered. Almost all approaches to PLD tacitly assume that this choice model is known precisely and is beyond suspicion. In practice, however, this is not the case. Typically the firm can only estimate the model from data that is obtained via conjoint analysis, wherein a small number of customers from the overall customer population is asked to either rate or choose from different hypothetical products. As such, the firm may face significant *uncertainty* in the choice model. This uncertainty can be of two types:

1. **Parameter uncertainty**. For a given parametric choice model that fits the data, the firm may be uncertain about the true parameter values (e.g., the partworths and the segment probabilities), due to the small sample size of the data. The concept of parameter uncertainty and the question of how to make decisions under parameter uncertainty have been well-studied in the operations research community in other applications (we survey some of the relevant literature in Section 3.2) but have received little attention in the context of product line decisions.

2. **Structural uncertainty**. For the given conjoint data, the firm may be uncertain about the right type of parametric model to use to describe the customer population (e.g., a first-choice model or a latent-class multinomial logit model). This type of uncertainty is conceptually different from parameter uncertainty, where the focus is restricted to a single type of model. The concept of structural uncertainty is a new one that was recently proposed and studied in [20] in the context of designing screening policies for prostate cancer under different models of the evolution of a patient's health. To the best of our knowledge,

this type of uncertainty has not been considered previously in the context of product line decisions.

Uncertainty is of significant interest because the optimal product line can change dramatically as one considers different parameter values or different structures of the choice model. Moreover, a product line that is designed for a particular choice model may result in very different revenues if the realized choice model is different from the planned one.

The issue of uncertainty becomes even more significant when one considers the nature of product line decisions. A firm's decision of which products to offer is one that is made infrequently; in most cases, the decision to produce a particular collection of products is one that commits the firm's manufacturing, marketing and operational resources and cannot be easily reversed or corrected. This relative lack of recourse, along with the strategic importance of the decision, underscores the need for a product line design approach that is immunized to the uncertainty in the underlying choice model.

In this chapter, we propose a new optimization approach for product line design that addresses uncertainty in customer choice. Rather than finding the product line that maximizes the revenue under a single, nominal choice model, we propose a robust optimization approach where we specify a set of possible models – the *uncertainty set* – and find the product line that optimizes the worst-case expected revenue, where the worst-case is taken over all of the models in the uncertainty set. We make the following contributions:

1. We propose a new type of PLD problem that accounts for choice model uncertainty using robust optimization. Our approach is flexible in that it is compatible with many existing, popular choice models – such as the first-choice model, the latent class multinomial logit (LCMNL) model and the hierarchical Bayes mixture multinomial logit (HB-MMNL) model – and we propose different types of uncertainty sets that account for both parametric and structural uncertainty. We also discuss how to trade-off nominal and worst-case revenue by optimizing

a weighted combination of the two types of revenues.

2. We demonstrate the value of our approach through computational experiments using a real conjoint data set. In particular:

   - We consider parameter uncertainty in the first-choice model, the LCMNL model and the HB-MMNL model. We show that the nominal product line under each of these types of models is highly vulnerable to parameter uncertainty, and that the robust product line provides a significant edge in the worst case. For example, for the LCMNL model with $K = 3$ customer segments, the revenue of the nominal product line drops by over 10% in the worst-case, while the robust product line leads to worst-case revenues that are 4% higher than the nominal product line.

   - We consider structural uncertainty through two examples. In the first example, we consider an uncertainty set of LCMNL models where the models vary in terms of the number of segments K; here, we find that the nominal product line under any one LCMNL model can result in a worst-case revenue that is lower by 3 to 7%, while the robust product line outperforms the nominal product lines in the worst-case by 2 to 4%. In the second example, we consider an uncertainty set that consists of LCMNL models and the FC model. Here we find that the LCMNL models and the FC model lead to strikingly different product lines, and the product line that is optimal for each individual model is highly suboptimal under the other models. In contrast, the robust product line that accounts for all of the structurally distinct models outperforms each nominal product line under the worst-case model.

The rest of the chapter is organized as follows. In Section 3.2, we review the existing literature on product line design. In Section 3.3, we describe our PLD approach to address uncertainty in customer choice. In Section 3.4, we present extensive computational evidence that both highlights the need to account for uncertainty and the

benefits of adopting our approach to address uncertainty. Finally, in Section 3.5, we conclude the chapter and discuss some possible directions for future work.

3.2 Literature review

To date, significant research has been conducted in solving the PLD problem. The majority of approaches to the PLD problem model it as an integer optimization problem and assume a first-choice model of customer behavior; examples include [98], [54], [71], [42], [65] and [43]. The approaches differ in whether the procedure selects the product line from a predefined set of products ("product space" procedures; examples include [54, 71, 42, 43]) or whether the procedure directly prescribes the attributes and is not restricted to a predefined set ("attribute space" procedures; examples include [98, 65]). The approaches also differ in the solution method. For example, [98] and [71] solve their respective PLD problems exactly as integer optimization problems, while [54] consider several heuristics including a greedy heuristic and [65] consider a dynamic programming-based heuristic. For a comprehensive comparison of heuristics for the nominal first-choice PLD problem, the reader is referred to the paper of [9].

Apart from the first-choice model, other product line research has considered probabilistic choice models, such as the multinomial logit (MNL) model, where customers do not deterministically select their highest-utility choice, but rather they randomly select from all of their choices; typically, options with higher utilities are selected more frequently. Examples of product line approaches built on probabilistic choice models include [33], [67], [85] and [86]. In [33], the authors consider the problem of selecting a product line from a finite set of products to maximize expected revenue under the MNL model. Although the problem belongs to the class of mixed-integer nonlinear optimization problems, which are typically very difficult to solve, the paper shows that due to the problem structure it is sufficient to solve the continuous relaxation to obtain an optimal solution to the original problem. [85] builds on this modeling approach to incorporate price discrimination for different customer segments, while [86] further extends the approach to more general attraction models. [67] consider a

share-of-surplus choice model, where the probability of a segment selecting a product is the ratio of the surplus of that product to the total surplus over all of the products. The problem they formulate is a mixed-integer nonlinear optimization problem which they solve with a combination of simulated annealing and steepest ascent. [67] also additionally address uncertainty in the utilities by modeling the uncertain utilities as random variables and applying a heuristic that attempts to optimize the expected profit. Outside of the academic literature, Sawtooth Software's Advanced Simulation Module (ASM) [84] allows analysts to perform simulations and to find product lines that optimize one of a number of objectives (such as market share or revenue) with respect to different choice models. These choice models include the first-choice model, the multinomial logit model and the "randomized first-choice" (RFC) model. The RFC model is a type of probabilistic choice model, wherein one simulates many customers who choose according to the first-choice rule with a randomly generated utility function; the product utilities are altered for each customer by perturbing the partworths ("attribute error") and then perturbing the utility of each product ("product error").

The approach we propose in this chapter differs from the extant PLD literature in two main ways. First, existing approaches to the PLD problem under the first-choice model and under probabilistic choice models assume that the choice model that describes the customer population is known precisely. For example, in the first-choice model, it is assumed that we precisely know the segments comprising the customer population along with their partworths and sizes. As discussed earlier, typically the parameter values that define the choice model may not be known with complete certainty. As we will show later, product lines that do not account for errors in the parameter values that define the choice model can lead to significantly lower revenues when the realized parameter values are different from their planned values. To the best of our knowledge, no other work has highlighted the importance of accounting for parameter uncertainty in PLD or proposed a PLD approach that directly accounts for this uncertainty. Closest in spirit to some of the analyses we conduct is part of the paper of [9], where the authors present a post-hoc test of

how robust the product lines are with respect to error in the partworth utilities. In this experiment, the firm observes partworths that are perturbed from the true partworths (to simulate measurement error) and designs the product line under the perturbed partworths; the authors show that in this setting, the realized revenues are 2-5% lower than their anticipated values due to this measurement error. In this chapter, we consider uncertainty more generally over a wide range of choice models (the first-choice model, the LCMNL model and the HB-MMNL model) and propose an approach that accounts for uncertainty upfront; as such, our work complements this existing set of results. With regard to the RFC model implemented in Sawtooth Software's ASM, we emphasize that the RFC model addresses *randomness* in choice in the same way that other probabilistic choice models do; attribute error is analogous to assuming a probability distribution over partworths (as in the LCMNL model, which assumes a discrete mixture distribution or the HB-MMNL model which assumes a multivariate normal mixture distribution) and product error is analogous to utility error in a random utility framework (as in the multinomial logit model, which assumes standard Gumbel errors, or the multinomial probit model, which assumes normally distributed errors). The RFC model does not address uncertainty in a worst-case way, as we do in this chapter.

Second, virtually all existing approaches to the PLD problem focus on a single parametric structure for the choice model. In contrast, our approach is also designed to accommodate structural uncertainty and account for *multiple* parametric structures. The motivation for this setting is that often we may not be able to isolate a single parametric model as the "best" model, but instead we may have a collection of models that are "good enough" and that could all plausibly describe how the customer population will respond to the product line. The danger of optimizing for a single model is that we do not take the other models into consideration: while the single model product line will guarantee the optimal revenue under the chosen model, it is not guaranteed to deliver good performance under any of the other models. Thus, in our approach, rather than using a single model, we account for all of the models by optimizing a product line with respect to worst-case revenue over the set of models.

In this way, the product line is able to guarantee good performance over a range of models, rather than only guaranteeing good performance for a single model. To the best of our knowledge, the idea of optimizing for a set of structurally distinct models has not been proposed in the PLD literature.

Outside of the PLD literature, our treatment of parameter uncertainty is closely aligned with the large body of work within the operations research community on robust optimization [15]. Robust optimization is a technique for optimization under uncertainty, where one wishes to solve an optimization problem that is affected by uncertainty either through its constraints or its objective function. Robust optimization has been successfully applied in a variety of domain areas to address parameter uncertainty, including inventory management [22], portfolio optimization [21, 51], power system operations [17], cancer treatment [26] and facility location decisions [6, 32]. The only application of robust optimization techniques to a problem involving parametric choice models that we know of is the paper of [82]. In this paper, the authors consider the problem of assortment optimization – a problem closely related to PLD – under the multinomial logit model with uncertain utilities. The authors develop theoretical results that show that the problem can be solved efficiently. Our work contrasts with this work in that it does not exploit specific structural properties of a specific robust optimization problem to propose a tailored solution approach, but rather, it proposes a generic approach that allows for robustness to be incorporated under a wide variety of choice models.

Our treatment of structural uncertainty was inspired by the paper of [20]. The paper considers the problem of selecting a prostate cancer screening strategy to maximize a patient's expected quality adjusted life years. This problem is made difficult by the existence of three distinct, yet clinically recognized, models of how a patient's health evolves under different screening strategies. To reconcile these different models, the authors propose optimizing a weighted combination of the worst-case patient outcome and the average outcome over the three models. The authors show that the resulting screening strategy leads to good outcomes under all three models and is superior to the single-model screening strategies in both its worst-case and average-case

performance. The results in this chapter complement those in [20] by showing that this idea can have significant impact in product line decisions.

3.3 Model

3.3.1 Nominal model

Let n be the number of different attributes of the product. We assume, for simplicity, that each attribute is a binary attribute, i.e., either the product possesses the attribute or not. (This assumption is without loss of generality, as attributes with more than two levels can be modeled by, for example, introducing a binary attribute for each level of the original attribute and requiring that the product only possesses one of the binary attributes.) A product is then a binary vector $\mathbf{a} = (a_1, a_2, \ldots, a_n)$. We let N denote the number of candidate products and we index these products from 1 to N; for $p \in \{1, \ldots, N\}$, $\mathbf{a}^p$ indicates the attribute vector that encodes product p.

We assume that the marginal revenue of product p is given by $r(p)$. To model the market, we assume that each customer makes exactly one choice: he either chooses one of the products in the product line, or he chooses a "no-purchase" option, which corresponds to not purchasing a product from the product line (e.g., because of the existence of a more desirable product by a competing firm). The purchasing behavior of customers is described by a choice model m; the quantity $m(i \mid S)$ denotes the probability that the customer selects product i when offered the product line $S \subseteq \{1, \ldots, N\}$. We use the index 0 to indicate the no-purchase option, so that $m(0 \mid S)$ denotes the probability that the customer does not purchase any of the products in S. The expected revenue of product line S is then given by

$$R(S; m) = \sum_{i \in S} m(i \mid S) \cdot r(i). \tag{3.1}$$

Let P be the number of products that are to compose the product line (the "width" of the product line). We assume that P is predefined by the firm and is *not* a decision

variable. The nominal product line design problem can then be stated as

$$\underset{S\subseteq\{1,\ldots,N\}:|S|=P}{\text{maximize}} R(S;m). \tag{3.2}$$

In words, this is the problem of finding a product line S consisting of P products from the universe of products $\{1,\ldots,N\}$ that leads to the highest possible expected revenue under the choice model m.

There are many possible choices for the choice model m; we present three examples below.

1. **First-choice**. We assume that there are finitely many customer types; let K denote the number of customer types. We assume that K is pre-defined by the firm and is *not* a decision variable. We let λ^k denote the probability that a random customer is of type k. We thus have that $\sum_{k=1}^{K}\lambda^k = 1$ and $\lambda^k \geq 0$ for all $k \in \{1,\ldots,K\}$. For each customer type $k \in \{1,\ldots,K\}$, we let $u^k(p)$ denote the utility of product $p \in \{1,\ldots,N\}$ and we let $u^k(0)$ denote the utility of the no-purchase option for type k. We assume that the utilities are unique: if $p \neq p'$, then $u^k(p) \neq u^k(p')$, i.e., two products cannot have the same utility. Such an assumption is helpful in ensuring that the first-choice model is well-defined. Given that the products are defined by discrete attributes, this assumption is not particularly restrictive.

   With these definitions, the choice probability $m(i \mid S)$ can be defined as

$$m(i \mid S) = \sum_{k=1}^{K} \lambda^k \cdot \mathbb{I}\left\{i = \arg\max_{j\in S\cup\{0\}} u^k(j)\right\}, \tag{3.3}$$

   and the expected revenue $R(S;m)$ is just

$$R(S;m) = \sum_{i\in S}\left(\sum_{k=1}^{K} \lambda^k \cdot \mathbb{I}\left\{i = \arg\max_{j\in S\cup\{0\}} u^k(j)\right\}\right)\cdot r(i). \tag{3.4}$$

   We assume (as is commonly done in conjoint analysis) that the utility of a

product p is a linear function of its attributes, that is,

$$u^k(p) = \sum_{i=1}^{n} u_i^k \cdot a_i^p \tag{3.5}$$

where u_i^k is the partworth utility of attribute i for customer type k. We will use $\mathbf{u}$ to denote the vector of attribute partworths, i.e., $\mathbf{u}^k = (u_1^k, \ldots, u_n^k)$.

2. **Latent class multinomial logit**. We now define the latent class multinomial logit (LCMNL) model. As in the first-choice model, we assume that there are K customer segments. A customer belongs to segment k with probability λ^k. Each product $p \in \{1, \ldots, N\}$ provides some utility $u^k(p)$ to customers in segment k and as before, we assume that $u^k(0)$ is the utility of the no-purchase option. The choice probability $m(i \mid S)$ is then given by

$$m(i \mid S) = \sum_{k=1}^{K} \lambda^k \cdot \frac{\exp(u^k(i))}{\sum_{i' \in S} \exp(u^k(i')) + \exp(u^k(0))} \tag{3.6}$$

and the expected revenue under m is given by

$$R(S; m) = \sum_{i \in S} \sum_{k=1}^{K} \lambda^k \cdot \frac{r(i) \cdot \exp(u^k(i))}{\sum_{i' \in S} \exp(u^k(i')) + \exp(u^k(0))}. \tag{3.7}$$

As in the first-choice model, we will assume that the utility $u^k(p)$ is a linear function of the attributes of product p (cf. equation (3.5)) and use $\mathbf{u}$ to denote the vector of attribute partworths, i.e., $\mathbf{u}^k = (u_1^k, \ldots, u_n^k)$.

3. **Mixture multinomial logit**. In the mixture multinomial logit (MMNL) model, we assume that customers choose according to an MNL model, where the partworth vector $\mathbf{u} = (u_1, \ldots, u_n)$ is distributed according to some mixture distribution F. The choice probabilities are then given by

$$m_F(i \mid S) = \int \frac{\exp(u(i))}{\sum_{i' \in S} \exp(u(i')) + \exp(u(0))} \, dF(\mathbf{u}), \tag{3.8}$$

where $u(i)$ is the utility of product i under the partworth vector $\mathbf{u}$. Note that

when F is a distribution with discrete support, the choice model reduces to the LCMNL model (3.6). It is common to assume that $\mathbf{u}$ is normally distributed with mean $\boldsymbol{\beta}$ and covariance matrix $\mathbf{V}$. Thus, given $\boldsymbol{\beta}$ and $\mathbf{V}$, the corresponding choice probability can be written as

$$m_{\boldsymbol{\beta},\mathbf{V}}(i \mid S) = \int \frac{\exp(u(i))}{\sum_{i' \in S} \exp(u(i')) + \exp(u(0))} \cdot \phi(\mathbf{u}; \boldsymbol{\beta}, \mathbf{V})\, d\mathbf{u}, \tag{3.9}$$

where $\phi(\cdot; \boldsymbol{\beta}, \mathbf{V})$ is the density function of a multivariate normal random variable with mean $\boldsymbol{\beta}$ and covariance matrix $\mathbf{V}$.

Assuming a mixture distribution that is a multivariate normal, there are two approaches to estimating $\boldsymbol{\beta}$ and $\mathbf{V}$. The first approach is based on expectation maximization (see [93, 92]), which produces point estimates of $\boldsymbol{\beta}$ and $\mathbf{V}$. The second approach, which is more popular in marketing science and is more widely used in practice, is based on a hierarchical Bayes (HB) formulation of the model [4, 78, 79]. In the HB approach, $\boldsymbol{\beta}$ and $\mathbf{V}$ are treated as random variables with prior distributions, and the goal is to compute the posterior distributions of $\boldsymbol{\beta}$ and $\mathbf{V}$ given the available choice data from a conjoint analysis of a group of respondents. One of the most common specifications of the parameter priors (see [4, 79]) is given by

$$\boldsymbol{\beta} \sim N(\bar{\boldsymbol{\beta}}, \alpha \mathbf{V}), \tag{3.10a}$$

$$\mathbf{V} \sim IW(v_0, \mathbf{V}_0), \tag{3.10b}$$

where $N(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ indicates a multivariate normal distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$ and $IW(\nu, \mathbf{W})$ indicates the inverse Wishart distribution with degrees of freedom ν and scale matrix $\mathbf{W}$. Typically the parameters of the priors of $\boldsymbol{\beta}$ and $\mathbf{V}$ are specified to ensure that the priors are relatively diffuse. In what follows, we will use HB-MMNL to denote the hierarchical Bayes MMNL model.

Letting Ψ denote the posterior joint probability density function of $(\boldsymbol{\beta}, \mathbf{V})$, it

is common to compute the posterior mean of the choice probability $m_{\boldsymbol{\beta},\mathbf{V}}(i \mid S)$ (i.e., by integrating $m_{\boldsymbol{\beta},\mathbf{V}}(i \mid S)$ over Ψ), and to use this as the choice probability:

$$m(i \mid S) = \int \int \frac{\exp(u(i))}{\sum_{i' \in S} \exp(u(i')) + \exp(u(0))} \cdot \phi(\mathbf{u}; \boldsymbol{\beta}, \mathbf{V}) \cdot \Psi(\boldsymbol{\beta}, \mathbf{V}) \; d\mathbf{u} \, d\boldsymbol{\beta} \, d\mathbf{V}.$$

For most choice models m, the nominal PLD model (3.2) is a difficult optimization problem. For example, with the first-choice model, it is known that problem (3.2) is NP-Hard [65]. However, there exist many heuristics for solving problem (3.2) that are able to find high quality solutions in practical runtimes [9]. One such heuristic is the "divide-and-conquer" heuristic [55]. In this heuristic, the product line is broken into groups of attributes, and the procedure sequentially optimizes each group of attributes to improve the objective function. We follow [9]'s implementation of this heuristic in treating each product as a group of attributes; thus, the procedure changes the product line one product at a time until the revenue cannot be improved any further. Algorithm 2 provides a pseudocode description of the heuristic. This heuristic is guaranteed to determine a locally optimal product line, but is not guaranteed to find a globally optimal product line. However, [9] showed that this heuristic is able to quickly deliver very high quality solutions in the first-choice PLD problem, where quality was measured relative to the true optimal solution.

3.3.2 Robust model

In the nominal PLD problem (3.2), we optimize the expected per-customer revenue with respect to the choice model m. In doing so, we tacitly assume that this choice model is known precisely; that is, the choice model m that we plan for in problem (3.2) is exactly the choice model that will be realized when the product line is offered to the market. In reality, however, we normally do not know the choice model precisely and m is thus only an estimate of the true probability distribution; in keeping with the nomenclature of robust optimization, m is the *nominal* choice model. This uncertainty constitutes a significant risk to the firm. If the nominal choice model m is not accurate and the probability distribution that is realized is different from m, then

Algorithm 2 Divide-and-conquer heuristic for the nominal PLD problem (3.2).

Require: Choice model m, width of product line P, initial product line S
  Set **isLocalOptimal** = **false**
  while $\neg$ **isLocalOptimal** **do**
    for all $p \in S$ **do**
      Set $\mathcal{S}_p = \{S' \mid S' = S \cup \{p'\} \setminus \{p\}$ for some $p' \in \{1, \ldots, N\} \setminus S\}$
      Set $S_p^* = \arg\max_{S \in \mathcal{S}_p} R(S; m)$
      if $R(S_p^*; m) > R(S; m)$ **then**
        Set $S = S_p^*$
        break
      end if
    end for
    if $R(S_p^*; m) \leq R(S; m)$ for each $p \in S$ **then**
      Set **isLocalOptimal** = **true**
    end if
  end while
  return S

the revenue that is realized from the product line may be significantly lower than the anticipated revenue.

In this section, we propose an approach that directly accounts for the uncertainty in m. Rather than assuming that m is the true choice model, we can instead assume that we know a set of models $\mathcal{M}$, called the *uncertainty set*, which we believe contains the true choice model. This set may consist of models with the same parametric structure but different parameters, or models that have entirely different parametric structures; we discuss ways to construct $\mathcal{M}$ in more detail in Section 3.3.3. With $\mathcal{M}$ in hand, we proceed as follows: rather than solving a nominal optimization problem where we maximize the expected per-customer revenue with respect to the nominal m, we instead solve a *robust* optimization problem where we maximize the worst-case expected per-customer revenue, where the worst case is taken over all the possible choice models $\tilde{m}$ in the uncertainty set $\mathcal{M}$. Mathematically, the worst-case expected revenue of a product line S is defined as

$$R(S; \mathcal{M}) = \min_{\tilde{m} \in \mathcal{M}} R(S; \tilde{m}). \tag{3.11}$$

i.e., the worst-case expected per-customer revenue is the *minimum* of the expected

per-customer revenue over the possible choice models in $\mathcal{M}$. It is helpful to think of $\tilde{m}$ as being controlled by an adversary ("nature") that wishes to reduce the expected per-customer revenue that the decision maker garners; given the freedom to choose any $\tilde{m}$ from $\mathcal{M}$, nature will select the one that makes the expected per-customer revenue the lowest. For further background in robust optimization, the reader is referred to the review paper of [15].

The robust product line design problem can then be defined as

$$\underset{S\subseteq\{1,\ldots,N\}:|S|=P}{\text{maximize}} R(S;\mathcal{M}). \tag{3.12}$$

Note that, like the nominal PLD problem (3.2), the robust PLD problem (3.12) is still in general a difficult problem to solve to provable optimality. However, many of the same heuristics that are available for solving the nominal PLD problem (3.2) can be used to solve this problem. In particular, we can use the divide-and-conquer heuristic (Algorithm 2) to solve the problem, where we replace evaluations of the nominal objective function $R(\cdot;m)$ with evaluations of the worst-case objective function $R(\cdot;\mathcal{M})$.

Before we describe the possible forms of the uncertainty set $\mathcal{M}$, it is worthwhile to highlight three important aspects of the robust model. First, we would like to provide some further motivation for the objective in problem (3.12). The objective function (3.11) still considers the *expected* revenue of the product line, but unlike the nominal objective function (3.1), it considers the worst-case expected profit of the product line, where the worst-case is taken over the uncertainty set $\mathcal{M}$ of choice models. As discussed in Section 3.2, this contrasts with existing approaches to the PLD problem, all of which consider maximizing expected revenue or market share under a single, nominal choice model. The reason for taking our view is that each model in $\mathcal{M}$ represents a plausible outcome with regard to how the customer population might react to the product line and thus, for a product line S, the set $\{R(S;\tilde{m}) \mid \tilde{m} \in \mathcal{M}\}$ represents a set of possible outcomes in terms of the expected per-customer revenue. Different product lines will differ in terms of the set of possible revenue outcomes

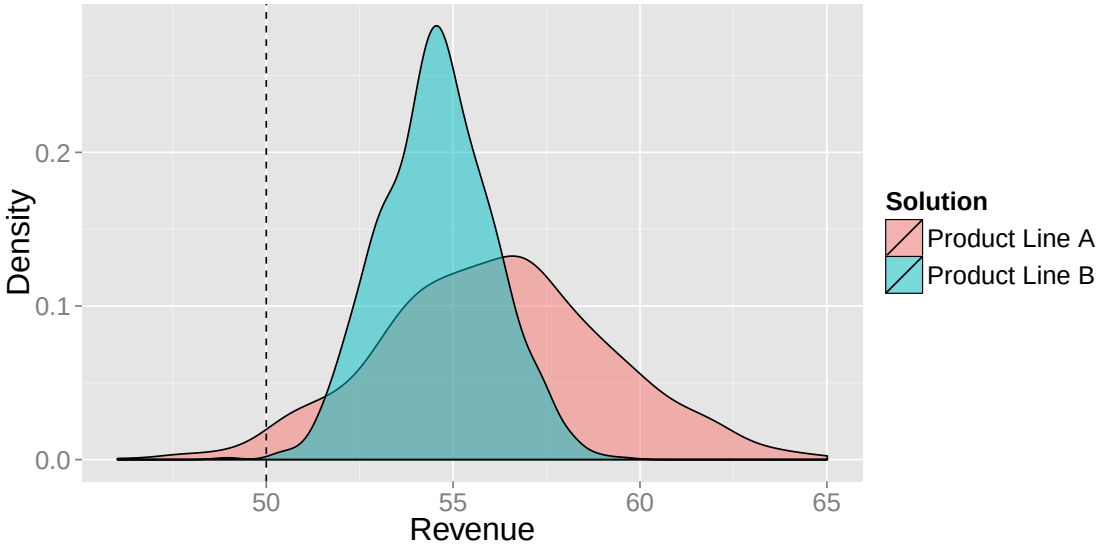

Figure 3-1: Hypothetical illustration of revenue distributions under two different product lines.

they induce. A low revenue outcome may have significant negative consequences for the firm's financial viability and its perceived performance. Thus, rather than ensuring a good average revenue outcome over the models in $\mathcal{M}$, the firm may wish to ensure a good revenue outcome under the least favorable model. To illustrate this, consider Figure 3-1, which displays the hypothetical expected per-customer revenue distributions induced by a set of choice models $\mathcal{M}$ for two different product lines. The dashed vertical line indicates a target value for the expected per-customer revenue. Here, we can see that while product line A ensures a better average performance over the models in $\mathcal{M}$ than product line B, there are many outcomes where the revenue under product line A is below the target revenue, as indicated by the much heavier tail below the target revenue. Thus, in this setting, product line B may be more desirable to the firm than product line A. This is the typical behavior observed when comparing nominal and robust product lines (here, product line A's revenue distribution is representative of that of a nominal product line, while product line B is representative of the robust product line).

The second aspect concerns how the solutions of the nominal problem (3.2) and the robust problem (3.12) relate to each other in terms of expected per-customer revenue.

Let S^{N} be an optimal collection of products for the nominal problem (3.2) and let S^{R} be an optimal collection of products for the robust problem (3.12) with uncertainty set $\mathcal{M}$. Then it follows that the worst-case expected per-customer revenue of the robust solution is always at least as good as the worst-case expected per-customer revenue of the nominal solution, that is,

$$R(S^{\mathrm{R}}; \mathcal{M}) \geq R(S^{\mathrm{N}}; \mathcal{M}).$$

To see this, observe that S^{N} is a feasible solution for problem (3.12); since S^{R} is an optimal solution for problem (3.12), it follows that S^{N} cannot have a higher value of $R(\cdot\,; \mathcal{M})$ as this would contradict the optimality of S^{R} for problem (3.12). A similar result follows in the other direction with respect to the nominal expected per-customer revenue: the nominal expected per-customer revenue of the nominal solution is always at least as good as the nominal expected per-customer revenue of the robust solution, that is,

$$R(S^{\mathrm{R}}; m) \leq R(S^{\mathrm{N}}; m).$$

In summary: the robust solution will perform better than the nominal solution in the worst case, while the nominal solution will perform better than the robust solution in the nominal case.

The third aspect of the robust model that is important to consider is the form and the size of the uncertainty set. As the size of the uncertainty set $\mathcal{M}$ increases, the minimization in (3.11) is taken over a larger set, and thus the corresponding solution of problem (3.12) will ensure the best worst-case performance over a larger set of possible values of $\tilde{m}$. However, as the uncertainty set increases in size, typically the performance of the robust solution in the *nominal* case degrades. Thus, there is a tradeoff in selecting the uncertainty set: if the set is too large and contains values of $\tilde{m}$ that are significantly different from the nominal value of m, then the robust solution will be protected against these values (relative to the nominal solution), but at the cost of performance under the nominal value of m. On the other hand, if the set is too small, then the robust solution will have better performance under the nominal

value of m, but will be vulnerable to extreme values of $\tilde{m}$.

3.3.3 Choices of the uncertainty set

We now discuss some ways that we may generate the uncertainty set $\mathcal{M}$.

1. **Finite set of variations of a single model**. Suppose that we fix a given type of model – for example, a latent-class multinomial logit model with K classes. We may then generate B different versions of the same type of model, leading to the uncertainty set

$$\mathcal{M} = \{m_1, \ldots, m_B\}.$$

   These variations of could be generated in a number of ways:

   (a) **Bootstrapping**. We may generate B bootstrapped samples of the conjoint data by sampling with replacement from the respondents in the data set. We can then run the estimation procedure of our desired model class on the choice data of each bootstrapped sample, thus generating B different models.

   (b) **Posterior sampling**. Bayesian models like HB will furnish us with a posterior distribution of a parameter $\boldsymbol{\theta}$ that specifies the model. For example, in the MMNL model in equation (3.9), the underlying model is specified by the mean and covariance matrix pair $(\boldsymbol{\beta}, \mathbf{V})$. In a Bayesian setting, the parameter $\boldsymbol{\theta}$ is unknown, and our uncertainty in this parameter is captured in the posterior distribution of $\boldsymbol{\theta}$ given the available choice data. To form $\mathcal{M}$, we may therefore take B independent samples from the posterior distribution of $\boldsymbol{\theta}$ – say, $\boldsymbol{\theta}^1, \ldots, \boldsymbol{\theta}^B$ – and set each m_b to be the choice model with parameter $\boldsymbol{\theta}^b$.

   (c) **Multiple models from the estimation procedure**. It may be the case that the estimation procedure generates multiple models from the same data. For example, the expectation maximization (EM) procedure [40],

which is the typical estimation method of choice for latent-class multinomial logit models and other popular discrete choice models, may converge to B different local minima given different initial starting points that are similar in their log likelihood.

2. **Continuous set of variations of a single model**. Rather than considering a finite collection of B different versions of the same model, we may want to consider sets where one or more parameters that define the model vary continuously within some set. Letting $\boldsymbol{\theta}$ denote the parameter, Θ denote the set of possible values of $\boldsymbol{\theta}$ and $m_{\boldsymbol{\theta}}$ denote the model that corresponds to $\boldsymbol{\theta}$, $\mathcal{M}$ will be defined as

$$\mathcal{M} = \{m_{\boldsymbol{\theta}} \mid \boldsymbol{\theta} \in \Theta\}$$

and correspondingly, $R(S; \mathcal{M})$ will be defined as

$$\begin{aligned} R(S; \mathcal{M}) &= \min_{\tilde{m} \in \mathcal{M}} R(S; \tilde{m}) \\ &= \min_{\boldsymbol{\theta} \in \Theta} R(S; m_{\boldsymbol{\theta}}). \end{aligned} \tag{3.13}$$

In the previous choice of the uncertainty set $\mathcal{M}$, where the uncertainty set $\mathcal{M}$ consists of finitely many models, the worst-case revenue $R(S; \mathcal{M})$ can be easily computed: simply compute $R(S; m)$ under each of the finitely many models m in $\mathcal{M}$ and take the minimum of this set of B values. In contrast, when the uncertainty set $\mathcal{M}$ consists of infinitely many models, the function $R(S; \mathcal{M})$ may not be easy to evaluate. As shown above in equation (3.13), the difficulty of computing $R(S; \mathcal{M})$ depends on how hard it is to optimize $R(S; m_{\boldsymbol{\theta}})$ as a function of the parameter $\boldsymbol{\theta}$ over the set of possible parameter values Θ.

One broad case where $R(S; \mathcal{M})$ is easy to compute is when $R(S; m_{\boldsymbol{\theta}})$ is a linear function of $\boldsymbol{\theta}$ and Θ is a polyhedron (that is, a set defined by finitely many linear equalities and inequalities on $\boldsymbol{\theta}$). In this case, $R(S; \mathcal{M})$ can be computed by solving a linear optimization problem, which is a theoretically tractable problem; this can also be done efficiently in practice provided that the dimension of $\boldsymbol{\theta}$ and

the number of constraints defining Θ are not too large. An example of such a case is when we consider uncertainty in customer type probabilities in the first-choice model. More concretely, we assume that the customer type probability distribution $\tilde{\boldsymbol{\lambda}}$ is uncertain and belongs to some polyhedral set Λ. Letting $m_{\tilde{\boldsymbol{\lambda}}}$ denote the first-choice model with probability distribution $\tilde{\boldsymbol{\lambda}}$, we can write

$$\begin{aligned} R(S;\mathcal{M}) &= \min_{\tilde{\boldsymbol{\lambda}}\in\Lambda} R(S;m_{\tilde{\boldsymbol{\lambda}}}) \\ &= \min_{\tilde{\boldsymbol{\lambda}}\in\Lambda} \sum_{k=1}^{K} \tilde{\lambda}^k \left(\sum_{i\in S} \mathbb{I}\{i = \arg\max_{j\in S\cup\{0\}} u^k(j)\} \cdot r(i) \right). \end{aligned} \tag{3.14}$$

In equation (3.14), we can see that the function being minimized is linear in $\boldsymbol{\lambda}$ and thus computing $R(S;\mathcal{M})$ amounts to solving a linear optimization problem. We now present two choices for the parameter uncertainty set Λ:

(a) **Box uncertainty set.** We assume that for each segment k, $\tilde{\lambda}^k$ is restricted to lie between $\underline{\lambda}^k$ and $\overline{\lambda}^k$. The *box* uncertainty set Λ_{box} is then given by

$$\Lambda_{\text{box}} = \left\{ \tilde{\boldsymbol{\lambda}} \in \mathbb{R}^K \;\middle|\; \begin{array}{l} \underline{\lambda}^k \le \tilde{\lambda}^k \le \overline{\lambda}^k, \; \forall\, k \in \{1,\ldots,K\}, \\ \mathbf{1}^T\tilde{\boldsymbol{\lambda}} = 1, \\ \tilde{\boldsymbol{\lambda}} \ge \mathbf{0}. \end{array} \right\}. \tag{3.15}$$

(b) **Aggregate deviation uncertainty set.** A limitation of the box uncertainty set is that the worst-case $\tilde{\boldsymbol{\lambda}}$ – that is, the $\tilde{\boldsymbol{\lambda}}$ that achieves the worst-case revenue (3.14) – may be one where multiple $\tilde{\lambda}^k$'s simultaneously take their most extreme values (either $\underline{\lambda}^k$ or $\overline{\lambda}^k$). Such a $\tilde{\boldsymbol{\lambda}}$ may be highly unlikely. Consequently, the resulting robust solution will sacrifice performance on more likely $\tilde{\boldsymbol{\lambda}}$'s in order to be (unnecessarily) protected against such extreme values of $\tilde{\boldsymbol{\lambda}}$.

An alternative uncertainty set that may be more appropriate, then, is one that not only bounds how much each individual λ^k value deviates from its nominal value, but that also bounds the *aggregate* deviation of the λ^k values from their nominal values. The resulting uncertainty set is the

aggregate deviation uncertainty set, which is defined as

$$\Lambda_{\text{AD}} = \left\{ \tilde{\boldsymbol{\lambda}} \in \mathbb{R}^K \;\middle|\; \begin{array}{l} \displaystyle\sum_{k=1}^{K} |\tilde{\lambda}^k - \lambda^k| \leq \Gamma, \\ \underline{\lambda}^k \leq \tilde{\lambda}^k \leq \overline{\lambda}^k, \; \forall\, k \in \{1, \ldots, K\}, \\ \mathbf{1}^T \tilde{\boldsymbol{\lambda}} = 1, \\ \tilde{\boldsymbol{\lambda}} \geq \mathbf{0}. \end{array} \right\}, \tag{3.16}$$

where $\sum_{k=1}^{K} |\tilde{\lambda}^k - \lambda^k|$ is the aggregate deviation and Γ is a user-specified bound on this deviation.

In the context of the first-choice model, one may ask if it is possible to account for uncertainty in the partworths of the customer types through a continuous uncertainty set. While it is possible, it turns out to be rather difficult. To illustrate this difficulty, suppose that for each customer type k, the set $\mathcal{U}^k$ denotes the uncertainty set for the partworth vector $\mathbf{u}^k$. Let $m_{\mathbf{u}^1,\ldots,\mathbf{u}^K}$ denote the choice model corresponding to the partworth vectors $\mathbf{u}^1, \ldots, \mathbf{u}^K$. The worst-case expected revenue is then

$$\begin{aligned} R(S;\mathcal{M}) &= \min_{(\tilde{\mathbf{u}}^1,\ldots,\tilde{\mathbf{u}}^K) \in \mathcal{U}^1 \times \cdots \times \mathcal{U}^K} R(S; m_{\tilde{\mathbf{u}}^1,\ldots,\tilde{\mathbf{u}}^K}) \\ &= \min_{(\tilde{\mathbf{u}}^1,\ldots,\tilde{\mathbf{u}}^K) \in \mathcal{U}^1 \times \cdots \times \mathcal{U}^K} \sum_{k=1}^{K} \lambda^k \left(\sum_{i \in S} \mathbb{I}\{i = \arg\max_{j \in S \cup \{0\}} \tilde{u}^k(j)\} \cdot r(i) \right) \\ &= \sum_{k=1}^{K} \lambda^k \min_{\tilde{\mathbf{u}}^k \in \mathcal{U}^k} \left(\sum_{i \in S} \mathbb{I}\{i = \arg\max_{j \in S \cup \{0\}} \tilde{u}^k(j)\} \cdot r(i) \right). \end{aligned} \tag{3.17}$$

To compute the worst-case expected revenue in equation (3.17), one must compute the minimum over each $\mathcal{U}^k$ of the expression

$$\sum_{i \in S} \mathbb{I}\{i = \arg\max_{j \in S \cup \{0\}} \tilde{u}^k(j)\} \cdot r(i).$$

The minimization for each customer type k is in general a difficult problem because the function being minimized is a non-convex function of $\mathbf{u}^k$. For this

reason, in our computational experiments in Section 3.4, we will not consider how to account for robustness under continuous partworth uncertainty.

3. **Finite set of multiple structurally distinct models**. Upon examining some conjoint data, we may estimate a handful of models that are structurally distinct but provide roughly the same quality of fit to the data – for example, a first-choice model m_{FC}, a latent class multinomial logit model with three segment m_{LC3} and a latent-class multinomial logit model with eight segments m_{LC8}. The uncertainty set is then just the collection of these three models:

$$\mathcal{M} = \{m_{\text{FC}}, m_{\text{LC3}}, m_{\text{LC8}}\}.$$

3.3.4 Trading off nominal and robust performance

As presented in Sections 3.3.1 and 3.3.2, the firm is faced with a choice between two alternative optimization paradigms: optimizing for the nominal choice model m or optimizing for the worst-case choice model among an uncertainty set $\mathcal{M}$. While the first paradigm is undesirable in that it completely ignores uncertainty, the second paradigm may also be unappealing in that it ignores what may be the most "likely" model that describes the customer population.

One way to bridge the two extremes is to consider robustness as a constraint. In this approach, one would find the product line that maximizes the nominal revenue subject to a constraint that the worst-case revenue is no lower than some predefined amount. Mathematically, this problem can be stated as

$$\underset{S\subseteq\{1,\ldots,N\}:|S|=P}{\text{maximize}} \quad R(S;m) \tag{3.18a}$$

$$\text{subject to} \quad R(S;\mathcal{M}) \geq \underline{R} \tag{3.18b}$$

where $\underline{R}$ is a desired lower bound on the worst-case revenue.

Although conceptually appealing, problem (3.18) comes with two practical challenges. First, as noted in Section 3.3.1, one can use heuristics to solve the robust PLD

problem (3.12) just as one can use them to solve the nominal PLD problem (3.2). However, to solve the constrained problem (3.18), existing PLD heuristics would need to be modified to ensure that the solution is feasible. With regard to the divide-and-conquer heuristic, a possible modification would be to consider a two-phase approach: in the first phase, we maximizes $R(S;\mathcal{M})$ to try to find a feasible solution; then, assuming we have found such a feasible solution, we maximize $R(S;m)$, taking care to only consider those solutions that are feasible (i.e., satisfying constraint (3.18b)) in each iteration.

Second, the firm may not have a single value of $\underline{R}$ in mind, but rather may be interested in seeing how the solution changes over a range of values of $\underline{R}$, in order to understand the trade-off between nominal and worst-case performance. An alternative way to proceed in this case is to consider optimizing a weighted combination of the nominal and worst-case revenues, as follows:

$$\underset{S\subseteq\{1,\ldots,N\}:|S|=P}{\text{maximize}} (1-\alpha)\cdot R(S;m) + \alpha \cdot R(S;\mathcal{M}). \tag{3.19}$$

Here, α is a weight between 0 and 1 that determines how much the objective emphasizes the worst-case revenue/de-emphasizes the nominal revenue. Note that unlike problem (3.18), problem (3.19) is an unconstrained problem that is amenable to existing PLD heuristics. By solving problem (3.19) for a range of values of α, it is possible to determine a Pareto efficient frontier of solutions that optimally trade-off worst-case revenue with nominal revenue.

3.4 Results

We now present the results of an extensive computational study with real conjoint data that illustrate (1) the need to account for uncertainty in product line design and (2) the benefits from adopting the approaches we prescribe. We begin by providing the background on the problem data in Section 3.4.1. We then present the results of several different experiments, which we summarize below:

- In Section 3.4.2, we consider how to account for uncertainty in customer type probabilities/segment sizes in the first-choice model. We show that the revenue of the nominal product line can deteriorate quite significantly in the presence of uncertainty, while the robust product line outperforms the nominal product line when both are exposed to moderate to high levels of uncertainty.

- In Section 3.4.3, we consider how to account for uncertainty in the LCMNL model by constructing the uncertainty set using bootstrapping. We show that the nominal product line under the LCMNL model is extremely susceptible to uncertainty in the LCMNL model parameters and the robust product line provides an edge over the nominal product line in the worst-case.

- In Section 3.4.4, we consider how to account for uncertainty in the HB-MMNL model by constructing the uncertainty set using samples from the posterior distribution of the mean and covariance parameters. We consider nominal solutions based on two different MMNL models – one using the posterior expectation of the choice probabilities (cf. (3)) and one based on a point estimate of the mean and covariance parameters – and we show that each one is outperformed by the robust solution in the worst-case.

- In Section 3.4.5, we consider how to account for structural uncertainty within the LCMNL model. Here, the structural uncertainty comes from the number of segments K, which is unknown to the modeler. We show that when there is uncertainty in the right value of K, committing to any one value of K can result in worst-case revenue losses ranging from 3 to 7% if the true value of K is different. The robust product line, which does not assume a single value of K but protects against a range of values of K, improves on the worst-case performance of the nominal product lines corresponding to the same range of K values by an amount of approximately 2-4%.

- Finally, in Section 3.4.6, we consider structural uncertainty under multiple structurally distinct models. The model uncertainty set consists of the first-choice

model and two different LCMNL models. We show here that the three nominal solutions lead to significantly lower revenues when a model with a different structure is realized, with worst-case losses ranging from 23 to 37%. In contrast, the robust approach is able to identify a product line that provides essentially the same revenue under all three models, and improves on the worst-case revenue of each nominal product line's revenue by an amount ranging from 13 to 55%.

3.4.1 Background

For the experiments we conduct here, we use a real conjoint data set from the field test of [91]. The field test from which the data set is derived is concerned with understanding consumer preferences for a hypothetical laptop bag to be offered by Timbuk2 (Timbuk2 Designs Inc., San Francisco, CA, USA). The data set consists of responses from 330 respondents. The data set contains two relevant pieces of data:

1. **Pairwise comparisons**. As part of the conjoint study, each respondent was required to compare 16 pairs of hypothetical laptop bags and indicate which product was more preferred or whether the respondent was indifferent to the choice.

2. **Estimated partworth utilities for the first-choice model**. Using the pairwise comparison data, the data set provides estimates of each respondent's first-choice partworth vector using the analytic center method of [91]. We use these partworths in our study of parameter uncertainty under the first-choice model in Section 3.4.2 and our study of structural uncertainty in Section 3.4.6.

The hypothetical laptop bag has ten different attributes, including the price, whether the bag has a handle and the color of the bag. We use the same revenue structure as [9], which previously used the data set of [91] – in particular, we assume the same marginal incremental revenues of [9]. We also assume, as in [9], that the price varies from \$70 to \$100 in \$5 increments.

We assume that the firm is interested in offering $P = 5$ versions of the laptop bag. As in [9], we assume that the no-purchase option involves the customer selecting one of three alternative products offered by the competition: (1) a bare-bones bag that includes no optional features and that is priced at \$70; (2) a mid-range bag that has five of the nine non-price features and is priced at \$85; and (3) a high-end bag that has all of the features and is priced at \$100.

To solve each nominal and robust PLD problem, we execute the divide-and-conquer heuristic described as Algorithm 2 from ten randomly chosen starting points, with the appropriate objective function (either a nominal objective function $R(S; m)$ or a worst-case objective function $R(S; \mathcal{M})$), and use the solution with the best objective value.

Our code, with the exception of the HB-MMNL model, was implemented in the Julia technical computing language [24]. All LCMNL models were estimated using a custom implementation of the EM algorithm (see [92]). All HB-MMNL models were estimated using the `bayesm` package in R [77].

In the experiments that follow, we will focus on two useful metrics for quantifying the benefit of robustness. The first is the worst-case loss (WCL). The WCL is defined for a nominal model m, a nominal product line S^{N} and an uncertainty set $\mathcal{M}$ as

$$\mathrm{WCL}(S^{\mathrm{N}}, m, \mathcal{M}) = 100\% \times \frac{R(S^{\mathrm{N}}; m) - R(S^{\mathrm{N}}; \mathcal{M})}{R(S^{\mathrm{N}}; m)}.$$

The WCL measures how much the revenue of the nominal product line deteriorates when one passes from the nominal revenue under m to the worst-case revenue over $\mathcal{M}$; in other words, it measures how vulnerable the product line is to the worst-case model in $\mathcal{M}$.

The second metric we will consider is the relative improvement (RI) of the robust product line over the nominal product line. The RI is defined for a nominal product line S^{N}, a robust product line S^{R} and an uncertainty set $\mathcal{M}$ as

$$\mathrm{RI}(S^{\mathrm{R}}, S^{\mathrm{N}}, \mathcal{M}) = 100\% \times \frac{R(S^{\mathrm{R}}; \mathcal{M}) - R(S^{\mathrm{N}}; \mathcal{M})}{R(S^{\mathrm{N}}; \mathcal{M})}.$$

The RI measures how much the robust product line improves on the nominal product line in terms of the worst-case revenue, relative to the nominal product line's worst-case revenue.

3.4.2 Parameter robustness under the first-choice model

In this section, we demonstrate the importance of accounting for uncertainty in the first-choice model.

We begin by defining the nominal model. We use the form of the model given by equation (3.4), where we take each respondent in the data set to represent a customer type. Thus, our first-choice model has $K = 330$ customer types. We assume that each customer type is equiprobable, i.e., $\lambda^k = 1/K$. We use the estimated partworths from [91] to form the utility function $u^k(\cdot)$ of each customer type $k \in \{1, \ldots, K\}$.

To model uncertainty, we will assume that the true probability distribution $\boldsymbol{\lambda}$ lies in a box uncertainty set Λ_{box} (cf. equation (3.15)) where the lower and upper bars $\underline{\boldsymbol{\lambda}}$ and $\overline{\boldsymbol{\lambda}}$ are parametrized by a scalar $\epsilon \geq 0$ in the following way:

$$\underline{\lambda}^k = (1 - \epsilon) \cdot \frac{1}{K}, \quad \forall\, k \in \{1, \ldots, K\}, \tag{3.20}$$

$$\overline{\lambda}^k = (1 + \epsilon) \cdot \frac{1}{K}, \quad \forall\, k \in \{1, \ldots, K\}. \tag{3.21}$$

For a given ϵ, we indicate the corresponding box uncertainty set by $\Lambda_{\text{box},\epsilon}$. Note that in this setting, we can interpret $\Lambda_{\text{box},\epsilon}$ as a set of different weightings of the respondents. For example, with $\epsilon = 2$, each respondent can have a weight $\tilde{\lambda}^k$ as low as 0 and as high as $3/K$ (i.e., the respondent is weighted three times as much as in the nominal probability distribution $\boldsymbol{\lambda}$, where $\lambda^k = 1/K$). Such an uncertainty set could be used to guard against the possibility that some portion of the respondents are outliers and not representative of the overall customer population.

We then proceed as follows. We solve the nominal PLD problem (3.2) to obtain a nominal product line S^{N}. Then, for a given value of ϵ, we solve the robust PLD problem (3.12), where the set of models $\mathcal{M}$ is induced by the uncertainty set $\Lambda_{\text{box},\epsilon}$, to obtain the robust product line $S^{\text{R},\epsilon}$. For each ϵ, we compute the WCL of the nominal

Uncertainty set $\Lambda_{\text{box},\epsilon}$	$R(S^{\text{N}};\mathcal{M})$ (\$)	$R(S^{\text{R},\epsilon};\mathcal{M})$ (\$)	WCL (%)	RI (%)
$\epsilon = 0.0$	72.82	72.82	0.00	0.00
$\epsilon = 0.1$	71.92	71.92	1.23	0.01
$\epsilon = 0.2$	71.02	71.09	2.38	0.10
$\epsilon = 0.5$	68.31	69.12	5.08	1.19
$\epsilon = 1.0$	63.80	67.12	7.83	5.19
$\epsilon = 2.0$	60.70	65.46	10.11	7.83
$\epsilon = 3.0$	57.67	64.88	10.91	12.50
$\epsilon = 4.0$	55.46	64.47	11.47	16.24

Table 3.1: Worst-case loss of nominal solution and relative improvement of robust solution over nominal solution for varying values of ϵ.

product line S^{N} and the RI of the robust product line corresponding to ϵ over the one nominal product line.

Table 3.1 shows how WCL and RI vary for values of $\epsilon \in \{0.1, 0.2, 0.5, 1, 2, 3, 4\}$. We can see from this table that for small amounts of uncertainty ($\epsilon < 0.5$), the revenue that is garnered under the nominal product line deteriorates moderately (e.g., for $\epsilon = 0.2$, the worst-case loss is 2.38%), and the robust product line provides a slight edge. However, for moderate to large amounts of uncertainty ($\epsilon \geq 0.5$), the worst-case loss increases significantly and the robust product line provides an edge over the nominal product line that grows as the amount of uncertainty increases. For example, with $\epsilon = 2$, the worst-case loss is 10.11%, and the robust product line delivers a worst-case revenue that is 7.83% higher than that of the nominal product line.

3.4.3 Parameter robustness under the LCMNL model

We now consider parameter robustness under the LCMNL model. In this set of experiments, we proceed as follows. For a fixed number of customer classes K, we estimate the nominal LCMNL model m using the pairwise comparison data set from all 330 respondents. Then, we generate a family of B LCMNL models $m_1, \ldots, m_B$ by bootstrapping. Each bootstrapped model is estimated from a bootstrapped data set that is generated by randomly sampling 330 respondents with replacement from

Num. segments K	$R(S^{\mathrm{N}}; m)$ (\$)	$R(S^{\mathrm{N}}; \mathcal{M})$ (\$)	$R(S^{\mathrm{R}}; \mathcal{M})$ (\$)	WCL (%)	RI (%)
$K = 1$	67.97	64.15	64.49	5.62	0.53
$K = 2$	66.40	60.03	60.77	9.59	1.22
$K = 3$	65.34	58.06	60.49	11.14	4.17
$K = 4$	64.87	59.40	59.61	8.43	0.36
$K = 5$	64.90	55.86	58.82	13.93	5.30
$K = 6$	66.19	55.50	58.50	16.16	5.42
$K = 7$	64.87	56.22	58.85	13.34	4.68
$K = 8$	67.02	51.60	58.15	23.02	12.70
$K = 9$	65.14	54.54	57.96	16.28	6.27
$K = 10$	65.49	50.62	57.62	22.70	13.82

Table 3.2: Comparison of nominal and worst-case revenues for LCMNL model under bootstrapping for $K \in \{1, \ldots, 10\}$.

the original set of 330 respondents. Note that by applying this procedure, the bootstrapped models effectively allow us to account for the uncertainty in the segment probabilities $\lambda^1, \ldots, \lambda^K$ and the segment-specific partworth vectors $\mathbf{u}^1, \ldots, \mathbf{u}^K$ jointly. We solve the nominal PLD problem (3.2) with the nominal model m to obtain the nominal product line S^{N}. We set $\mathcal{M} = \{m_1, \ldots, m_B\}$ and solve the robust PLD problem (3.12) with $\mathcal{M}$ to obtain the robust product line S^{R}.

We consider values of $K \in \{1, \ldots, 10\}$. We consider $B = 100$ bootstrapped models. For each estimation (for the nominal model and the B bootstrapped models), we run the EM algorithm from five different randomly generated starting points, and use the model with the highest log likelihood.

Table 3.2 compares the nominal revenues and the worst-case revenues over $\mathcal{M}$ of the two product lines for each value of K. We can see that under the worst-case model from the bootstrapped collection of models, the realized revenue can deteriorate significantly; for example, with $K = 5$ segments, the expected per-customer revenue is \$64.90 in the nominal case and \$55.86 in the worst-case, which is a loss of over 13%. Furthermore, we can see that in the worst-case, the robust product line is able to offer a significant improvement over the nominal product line, ranging from 0.36% ($K = 4$) to as much as 13.82% ($K = 10$).

To visualize the variability of revenues under each product line, Figure 3-2 plots a smoothed histogram of the revenue under the nominal and robust product lines

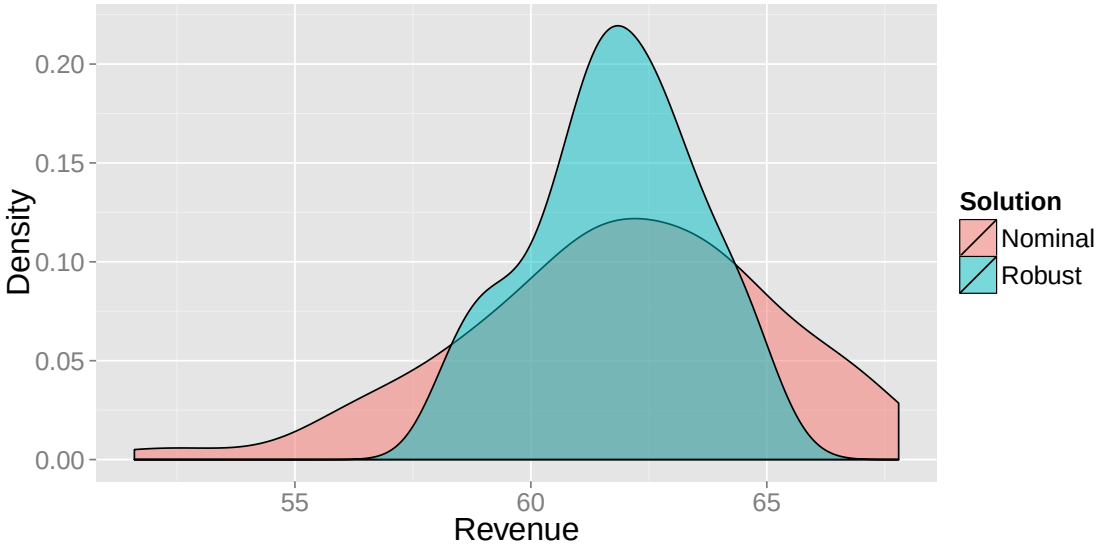

Figure 3-2: Plot of revenues under nominal and robust product lines under bootstrapped LCMNL models with $K = 8$.

for $K = 8$. The revenue distribution is formed by the M bootstrapped models in $\mathcal{M}$. We can see that the mean of the robust distribution is less than the mean of the nominal distribution, but the robust distribution has a lighter tail to the left and is more concentrated around its mean. Thus, if we believe that the bootstrapped models in $\mathcal{M}$ are all models that could be realized, then the robust product line will exhibit less risk than the nominal product line.

One interesting insight that emerges from Table 3.2 is that the relative improvement is generally higher with higher values of K. This makes sense, as the estimated parameter values of more complex models will be more sensitive to the underlying data used to estimate the model. By bootstrapping the data, there will be more variability in the family of models $\mathcal{M}$, and the robust product line S^{R} may therefore offer a greater edge over the nominal product line S^{N} in the worst-case. At the same time, in practice, more complex models (LCMNL models with higher values of K) may offer a better fit to the data than simpler models (LCMNL models with lower values of K). Therefore, robustness will become more desirable if we believe that we should use a large number of customer classes to describe our customer population.

We conclude our study of parametric robustness under the LCMNL model by

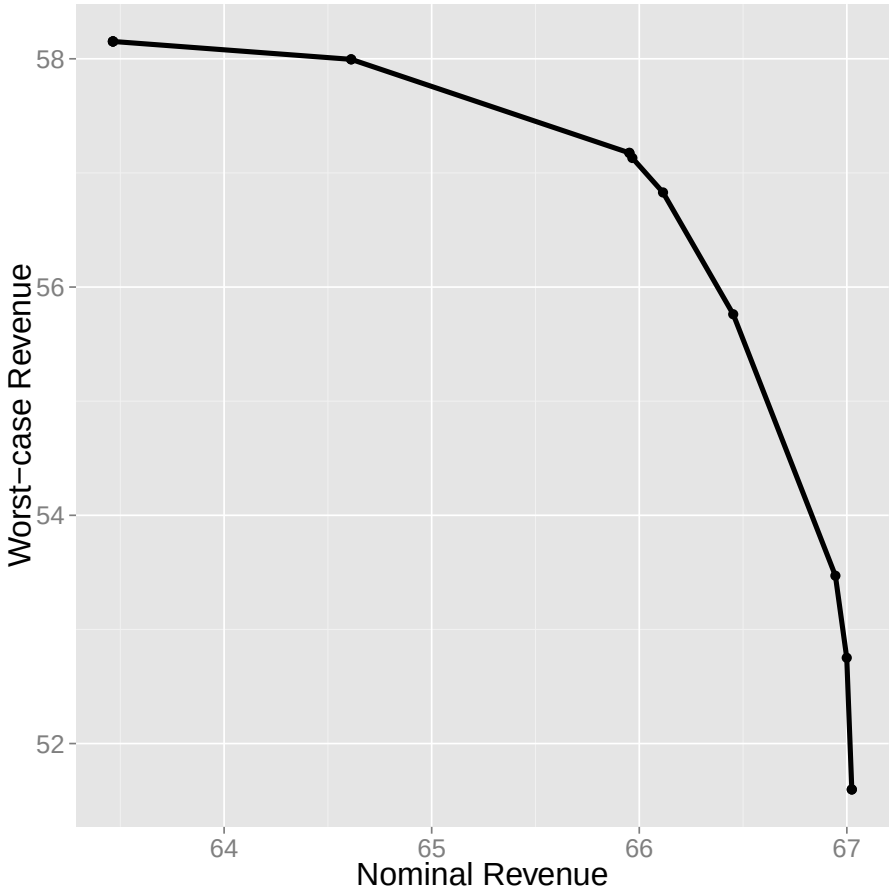

Figure 3-3: Plot of approximate Pareto efficient frontier of solutions that trade-off nominal revenue and worst-case revenue under the bootstrapped uncertainty set $\mathcal{M}$ for $K = 8$.

illustrating the trade-off between nominal and worst-case revenues using the weighted combination approach described Section 3.3.4. To demonstrate this, we focus on the LCMNL model with $K = 8$ and solve problem (3.19) for α values in

$$\{0.01, 0.02, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 0.95, 0.98, 0.99\}.$$

Figure 3-3 plots each solution of each α value as a point in two-dimensional space, where the x-axis indicates the nominal revenue of the solution and the y-axis indicates the worst-case revenue over $\mathcal{M}$ of the solution; the points are joined together by straight lines to form an approximate Pareto efficient frontier of solutions that are undominated in both nominal and worst-case revenue.

From Figure 3-3, we can see that there is a wide range of solutions between the robust solution (top-left end of frontier) and nominal solution (bottom-right end of frontier) and that the frontier suggests some profitable trade-offs. For example, we can see that there exist solutions that are very similar to the nominal solution in terms of nominal revenue but whose worst-case revenues are considerably higher. The main takeaway from this plot is that by considering problem (3.19), the firm can evaluate the range of solutions that are efficient with regard to nominal and worst-case revenue and decide which point on the frontier is most desirable to them.

3.4.4 Parameter robustness under the HB-MMNL model

In this section, we consider how to incorporate robustness under the HB-MMNL model using posterior sampling. We use the prior specification in equation (3.10) for the priors of the mean $\boldsymbol{\beta}$ and covariance matrix $\mathbf{V}$. We obtain B independent samples $(\boldsymbol{\beta}^1, \mathbf{V}^1), \ldots, (\boldsymbol{\beta}^B, \mathbf{V}^B)$ from the posterior distribution of $(\boldsymbol{\beta}, \mathbf{V})$. We obtain these samples through Markov chain Monte Carlo (MCMC), using the `bayesm` package in R [77]. We ran the MCMC procedure for 100,000 iterations with a burn-in period of 50,000 iterations. Due to the high autocorrelation in the draws of the parameter values, we thin the remaining draws to only retain every 200th draw. The last $B = 100$ of the thinned draws of $(\boldsymbol{\beta}, \mathbf{V})$ are used to form the set $\{(\boldsymbol{\beta}^1, \mathbf{V}^1), \ldots, (\boldsymbol{\beta}^B, \mathbf{V}^B)\}$. We use the default values provided by `bayesm` for the prior parameters in equation (3.10).

Ideally, we would like to consider the uncertainty set $\mathcal{M}$ that consists of $m_{\boldsymbol{\beta}^b, \mathbf{V}^b}$, as defined in equation (3.9), for each sample $b \in \{1, \ldots, B\}$. Unfortunately, each choice model of the form $m_{\boldsymbol{\beta}, \mathbf{V}}$ requires the evaluation of an integral of a multinomial logit choice probability with respect to a multivariate normal density, which cannot be computed in closed form. Thus, for each model $m_{\boldsymbol{\beta}^b, \mathbf{V}^b}$, we approximate the integral by drawing $T = 100$ samples from the multivariate normal density with mean $\boldsymbol{\beta}^b$ and covariance matrix $\mathbf{V}^b$; denoting the approximate choice model by $\hat{m}_{\boldsymbol{\beta}^b, \mathbf{V}^b}$, the choice

probability of an option i given the product line S is then just

$$\hat{m}_{\boldsymbol{\beta}^b,\mathbf{V}^b}(i \mid S) = \sum_{t=1}^{T} \frac{1}{T} \frac{\exp(u^{b,t}(i))}{\sum_{i' \in S} \exp(u^{b,t}(i')) + \exp(u^{b,t}(0))},$$

where $u^{b,t}(\cdot)$ is the utility function corresponding to the tth sample of the partworth vector $\mathbf{u}$ from the multivariate normal distribution corresponding to the bth posterior sample. Our final uncertainty set is thus

$$\mathcal{M} = \{\hat{m}_{\boldsymbol{\beta}^1,\mathbf{V}^1}, \ldots, \hat{m}_{\boldsymbol{\beta}^B,\mathbf{V}^B}\}.$$

We consider two types of nominal models. The first is the approximate posterior expectation (PostExp) MMNL choice model, which is defined as

$$\begin{aligned} m_{PostExp}(i \mid S) &= \frac{1}{B} \cdot \sum_{b=1}^{B} \hat{m}_{\boldsymbol{\beta}^b,\mathbf{V}^b}(i \mid S) \\ &= \frac{1}{BT} \sum_{b=1}^{B} \sum_{t=1}^{T} \frac{\exp(u^{b,t}(i))}{\sum_{i' \in S} \exp(u^{b,t}(i')) + \exp(u^{b,t}(0))}, \end{aligned}$$

The choice probabilities produced by $m_{PostExp}$ can be viewed as approximations of the true posterior expected choice probabilities given in equation (3). There are two levels of approximation: the integral over the posterior density of $\boldsymbol{\beta}$ and $\mathbf{V}$ is approximated by the average of B samples from that posterior density, while the inner integral for each posterior sample is approximated by the average of T samples from the corresponding multivariate normal distribution.

The second nominal model we will consider is the approximate point estimate MMNL choice model. Here, we obtain the approximate posterior mean of the mean parameter $\boldsymbol{\beta}$, given by $\boldsymbol{\beta}^* = (1/B) \cdot \sum_{b=1}^{B} \boldsymbol{\beta}^b$, and the covariance matrix, given by $\mathbf{V}^* = (1/B) \cdot \sum_{b=1}^{B} \mathbf{V}^b$. We then plug these point estimates into the definition of the MMNL model in equation (3.9). Since the integral that defines the equation (3.9) cannot be computed in closed form, we approximate the integral with $T = 100$ samples from the corresponding multivariate normal density. Letting $u^{*,t}(\cdot)$ denote

Model m	$R(S^{\mathrm{N},m}; m)$ (\$)	$R(S^{\mathrm{N},m}; \mathcal{M})$ (\$)	WCL (%)	RI (%)
$m_{PointEst}$	61.33	54.88	10.52	6.25
$m_{PostExp}$	62.44	56.89	8.89	2.50

Table 3.3: Comparison of solutions under nominal HB models $m_{PointEst}$ and $m_{PostExp}$ to robust solution under uncertainty set $\mathcal{M}$ formed by posterior sampling.

the utility function corresponding to the tth sample from the multivariate normal density with mean $\boldsymbol{\beta}^*$ and covariance matrix $\mathbf{V}^*$, we define the approximate point estimate (PointEst) MMNL choice model as

$$m_{PointEst}(i \mid S) = \frac{1}{T} \sum_{t=1}^{T} \frac{\exp(u^{*,t}(i))}{\sum_{i' \in S} \exp(u^{*,t}(i')) + \exp(u^{*,t}(0))}.$$

This approach of replacing the posterior distribution over $\boldsymbol{\beta}$ and $\mathbf{V}$ by a point estimate is sometimes referred to as the "plug-in" Bayes approach [78].

We optimize each of the resulting objective functions, $R(\cdot; \mathcal{M})$, $R(\cdot; m_{\mathrm{N},PostExp})$ and $R(\cdot; m_{\mathrm{N},PointEst})$, to obtain the product lines S^{R}, $S^{\mathrm{N},m_{PostExp}}$ and $S^{\mathrm{N},m_{PointEst}}$, respectively. For each nominal solution, we compute the worst-case loss (WCL) and the relative improvement (RI) of the robust solution under the uncertainty set $\mathcal{M}$.

Table 3.3 shows the nominal and worst-case revenue under $\mathcal{M}$ of the nominal product line under each of the two nominal HB models, as well as the WCL and RI for each nominal product line. The worst-case revenue of the robust product line is \$58.31. From this table, we can see that the nominal product lines obtained using the posterior expectation choice model $m_{PostExp}$ and the point estimate model $m_{PointEst}$ are highly susceptible to uncertainty as represented by the uncertainty set $\mathcal{M}$, which is informed by the posterior distribution of the partworth mean vector $\boldsymbol{\beta}$ and the covariance matrix $\mathbf{V}$. At the same time, the robust solution is able to improve the worst-case revenue under $\mathcal{M}$ significantly in each case (more than 6% for the point estimate model and by 2.5% for the posterior expectation model).

3.4.5 Structural robustness under different LCMNL models

One of the difficulties of using the LCMNL model in practice is that we do not know a priori what the right number of customer segments K is. To find the number of segments, one typically estimates the LCMNL model for a range of values of K and then selects the "best" model from this set of models. What is meant by "best" is usually a model that strikes a good balance between fit (as measured by the log likelihood) and complexity (as measured by the number of parameters). While there exist metrics such as the Akaike information criterion (AIC) [3], the Bayesian information criterion (BIC) [87] or the consistent Akaike information criterion (CAIC) [28] that one can use to quantify the quality of a model and to guide the selection of a best model (e.g., select the model with the best AIC value), these metrics are not infallible and can often lead to different choices of a best model. In the end, one may only be able to delineate a set of "good" models as opposed to a single "best" model.

To illustrate this with the conjoint data set of [91], we estimate the LCMNL model using the pairwise comparison data for different values of K in $\{1, 2, \ldots, 20\}$. Figures 3-4 and 3-5 display the AIC and CAIC, respectively, against the number of customer segments K. From Figure 3-5, we can see that the CAIC implies a choice of K of either 3 or 4. (The minimum CAIC is 6103.17 at $K = 3$; the second lowest is 6104.89 for $K = 4$.) On the other hand, from Figure 3-4, we can see that AIC continues decreasing as K increases beyond $K = 3$ until it begins to settle down, starting at approximately $K = 12$. Thus, based on the AIC, it would seem that $K = 12$ would be a more appropriate choice. Based on these plots, we could conclude that any value of K between 3 and 12 is a reasonable value to select.

We now demonstrate how robustness can be used to counteract our uncertainty in the number of segments in the LCMNL model. Given the plots of the AIC and CAIC in Figures 3-4 and 3-5, we may decide that the number of segments K should lie in the range between 3 and 12. Thus, letting $m_{\text{LC}K}$ denote the fitted LCMNL model with K customer segments, we would define our uncertainty set to be

$$\mathcal{M} = \{m_{\text{LC3}}, m_{\text{LC4}}, \ldots, m_{\text{LC12}}\}.$$

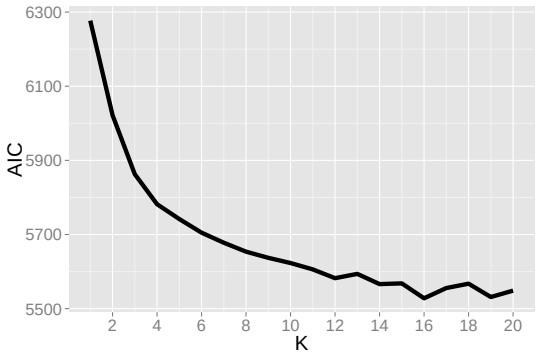

Figure 3-4: AIC for $K \in \{1, \ldots, 20\}$.

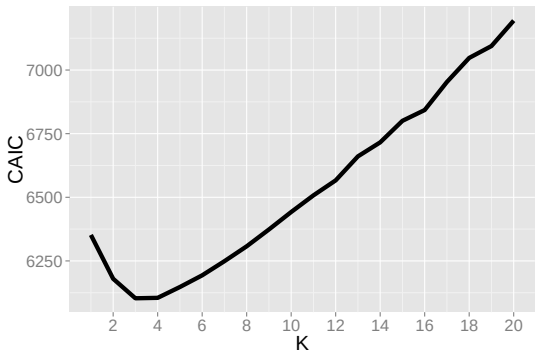

Figure 3-5: CAIC for $K \in \{1, \ldots, 20\}$.

We would then solve the robust problem (3.12) with $\mathcal{M}$ to obtain a robust product S^{R}. To compare this solution against the nominal approach, we solve the nominal problem (3.2) for each $m_{\mathrm{LC}K} \in \mathcal{M}$ to obtain a product line $S^{\mathrm{N},K}$ for each $K \in \{3, \ldots, 12\}$, and evaluate the worst-case revenue $R(S^{\mathrm{N},K}; \mathcal{M})$ for each such product line.

Table 3.4 provides the results of this comparison. The second column indicates each nominal product line's nominal revenue. The third column indicates each nominal product line's *worst-case* revenue over $\mathcal{M}$. The fourth column displays the relative loss from the nominal to the worst-case revenue, while the fifth column displays the improvement of the robust product line over each nominal product line with respect to the worst-case revenue. The worst-case revenue $R(S^{\mathrm{R}}; \mathcal{M})$ of the robust solution S^{R} is \$63.86.

From this table, we can see that the nominal product lines corresponding to $m_{\mathrm{LC3}}, \ldots, m_{\mathrm{LC12}}$ are quite susceptible to uncertainty. This is visible from the values of the worst-case loss; the worst-case loss ranges from 3.33% ($K = 7$) to as much as 7.01% ($K = 8$). At the same time, the robust solution S^{R} provides significant improvement over all of these nominal solutions in the worst-case. In particular, for the two models m_{LC3} and m_{LC12} suggested by the CAIC and AIC metrics, respectively, the relative improvements are 4.15% and 3.69%.

The main takeaway from this analysis is that robust optimization provides an alternative to model selection. When the data set does not clearly imply one single

Num. segments K	$R(S^{\mathrm{N},K}; m_{\mathrm{LC}K})$ (\$)	$R(S^{\mathrm{N},K}; \mathcal{M})$ (\$)	WCL (%)	RI (%)
$K = 3$	65.34	61.32	6.17	4.15
$K = 4$	64.87	61.56	5.10	3.74
$K = 5$	64.90	61.35	5.47	4.09
$K = 6$	66.19	62.58	5.46	2.05
$K = 7$	64.87	62.71	3.33	1.83
$K = 8$	67.02	62.33	7.01	2.46
$K = 9$	65.14	62.24	4.45	2.60
$K = 10$	65.49	62.33	4.82	2.46
$K = 11$	66.16	62.48	5.57	2.22
$K = 12$	65.80	61.59	6.41	3.69

Table 3.4: Comparison of nominal and worst-case revenues of product lines $S^{\mathrm{N},3}, \ldots, S^{\mathrm{N},12}$.

"best" model, but instead implies multiple "reasonably good" models, robustness provides a means of simultaneously optimizing against all of these models, rather than optimizing only against a single one.

3.4.6 Structural robustness under distinct models

In Section 3.4.5, we considered how robust optimization can be used to account for uncertainty in the number of segments in the LCMNL model. In this section, we take this idea a step further by using robust optimization to account for our uncertainty over *structurally distinct* models. In particular, in a given application, we may develop models that make very different assumptions. For example, we may estimate both a first-choice model, as in Section 3.4.2, and an LCMNL model as in Sections 3.4.3 and 3.4.5. As in Section 3.4.5, one model may not clearly stand out as the best model. At the same time, if we make a product line decision based on one of these models, the revenue that is garnered may be significantly lower than anticipated if the market responds according to one of the other models.

To demonstrate the value of robustness, we proceed as follows. We assume that after analyzing the conjoint data set, we identify the following set of models:

$$\mathcal{M} = \{m_{\mathrm{FC}}, m_{\mathrm{LC3}}, m_{\mathrm{LC12}}\}, \tag{3.22}$$

Product line	Nominal revenue ($) $R(\cdot; m_{\text{FC}})$	$R(\cdot; m_{\text{LC3}})$	$R(\cdot; m_{\text{LC12}})$	Worst-case revenue ($) $R(\cdot; \mathcal{M})$	WCL (%)	RI (%)
$S^{\text{N},m_{\text{FC}}}$	72.82	55.74	56.39	55.74	23.45	13.70
$S^{\text{N},m_{\text{LC3}}}$	40.98	65.34	61.32	40.98	37.28	54.64
$S^{\text{N},m_{\text{LC12}}}$	48.49	62.93	65.80	48.49	26.31	30.69
S^{R}	63.69	63.43	63.38	63.38	–	–

Table 3.5: Performance of nominal and robust product lines under the different models in $\mathcal{M}$ as well as the worst-case model.

where m_{FC} is the nominal first-choice model from Section 3.4.2 constructed using the estimated partworth vectors of all 330 respondents and m_{LC3} and m_{LC12} are the LCMNL models with $K = 3$ and $K = 12$ segments, respectively, estimated from the pairwise comparison data. For each model $m \in \mathcal{M}$, we solve the nominal PLD problem (3.2) to obtain the nominal product line $S^{\text{N},m}$. We compute the revenue of $S^{\text{N},m}$ under each model $m' \in \mathcal{M}$, as well as its worst-case revenue over $\mathcal{M}$. Then, we solve the robust PLD problem (3.12) to obtain the robust product line S^{R}, and compute the revenue of S^{R} under each model $m' \in \mathcal{M}$ as well as its worst-case revenue. We compute the worst-case loss (WCL) of each model m and the relative improvement (RI) of the robust product line S^{R} over the nominal product line for each model m.

Table 3.5 displays the nominal and worst-case revenues of each of the product lines. From this table, we can see that committing to a single model leads to significantly lower revenues under the other models. For example, if we assume the first-choice model m_{FC}, the nominal revenue is \$72.82, but this reduces to \$55.74 if the true model is the $K = 3$ segment LCMNL model, representing a loss of over 20%. If we instead assume the $K = 3$ LCMNL model, the nominal revenue is \$65.34, but this reduces to \$40.98 if the realized model is the first-choice model (a loss of over 35%). The WCL of all three nominal product lines is over 20%, indicating that in the worst-case, the revenue of each of these product lines deteriorates by over 20%.

In contrast to the nominal solution, the robust solution is able to achieve better worst-case performance, ranging from a 13.70% improvement (over the first-choice

Price	$100	$80	$95	$100	$70
Large size	■		■		
Red color			■		
Logo	■		■		
Handle	■		■	■	
PDA holder		■	■	■	
Cellphone holder		■	■	■	
Mesh pocket		■	■	■	
Velcro flap	■	■	■		■
Boot	■	■	■		

Figure 3-6: First-choice product line, $S^{\mathrm{N},m_{\mathrm{FC}}}$.

solution) to 54.64% (over the m_{LC3} solution). Moreover, the revenue of the robust solution is strikingly consistent over all three models, ranging from $63.38 to $63.69. Although the robust solution is not able to simultaneously achieve the best performance that is possible for each of the three models, it is able to achieve good performance for all three models and achieves better aggregate performance than the solution of each individual model. In this way, robust optimization allows us to incorporate all of the models that we deem acceptable, rather than just one.

It is also interesting to examine the product lines themselves. Figures 3-6, 3-7 and 3-8 display the nominal product lines for m_{FC}, m_{LC3} and m_{LC12}, respectively, while Figure 3-9 displays the robust product line. The first column indicates the attribute and each subsequent column represents one of the products; in a given column, a shaded cell indicates that the product has the corresponding attribute. For example, the fourth product in the first-choice product line is priced at $100, and has a handle, a PDA holder, a cellphone holder and a mesh pocket. For easier comparison, Figure 3-10 shows the competitive offerings, that is, the products comprising the no-purchase option in the same format.

From these figures, we can see that the three nominal product lines are very different in terms of the actual products. For example, the first-choice product line is structured in a way that it includes products that directly compete with the competitive offerings that comprise the no-purchase option. As a specific example, consider the third product, which is identical to the high-end bag (see Figure 3-10) in terms

Price	\$100	\$100	\$100	\$100	\$100
Large size		■		■	■
Red color					■
Logo	■		■		■
Handle	■	■	■	■	■
PDA holder					
Cellphone holder					
Mesh pocket		■			■
Velcro flap	■	■		■	■
Boot	■	■	■	■	■

Figure 3-7: LCMNL model with $K = 3$ product line, $S^{\mathrm{N},m_{\mathrm{LC3}}}$.

Price	\$100	\$100	\$100	\$80	\$100
Large size		■	■		■
Red color					■
Logo	■		■		■
Handle	■	■	■	■	■
PDA holder					■
Cellphone holder				■	■
Mesh pocket	■	■	■		■
Velcro flap		■	■	■	■
Boot	■	■	■		■

Figure 3-8: LCMNL model with $K = 12$ product line, $S^{\mathrm{N},m_{\mathrm{LC12}}}$.

Price	\$100	\$100	\$95	\$100	\$100
Large size	■		■	■	
Red color			■		
Logo	■		■		■
Handle	■	■	■	■	■
PDA holder			■		
Cellphone holder			■		
Mesh pocket	■	■	■	■	
Velcro flap	■		■	■	
Boot		■	■	■	■

Figure 3-9: Robust product line, S^{R}.

Price	$70	$85	$100
Large size			
Red color			
Logo			
Handle			
PDA holder			
Cellphone holder			
Mesh pocket			
Velcro flap			
Boot			

Figure 3-10: Competitive products.

of the non-price features, but is priced at $95 instead of $100 and is thus more attractive. In contrast to the first-choice product line, the LCMNL product lines have products that are different in terms of features and in general are priced higher. This difference in the product lines under the three models manifests itself in the results of Table 3.5, which shows how each nominal product line is suboptimal under one of the other two nominal models.

Comparing the nominal product lines to the robust product line in Figure 3-9, we can see that the robust product line contains three products found in the nominal single-model product lines. Specifically, the third product in the first-choice is identical to the third product in the robust product line; the second product in the LCMNL $K = 3$ product line and the fourth product in the robust product line are identical; and the third product in the LCMNL $K = 3$ product line and the fifth product in the robust product line are identical. Furthermore, the first and second product of the robust product line differ by one attribute from the third and first product, respectively, of the LCMNL $K = 12$ product line. Thus, the robust product line could be viewed as effectively "blending" the three single-model product lines. This characteristic of the robust product line agrees with our intuition: by using products that are identical or very similar to products in the individual single-model product lines, it seems reasonable to expect that the robust product line should achieve good performance over all three models.

3.5 Conclusion

In this chapter, we have extended the familiar nominal product line design problem to account for parameter uncertainty and structural uncertainty by adopting a robust optimization approach. With regard to parameter uncertainty, we showed through a number of numerical experiments using a real conjoint dataset that parameter uncertainty under three different widely used models has the potential to greatly reduce realized revenues. We further showed how, using the approach we have proposed, it is possible to guard against uncertainty and to limit its impact on revenues. With regard to structural uncertainty, we demonstrated that different model structures can lead to significantly different product line decisions, and that each such product line decision may be highly suboptimal for a different model structure. We then showed how optimizing against the worst-case revenue over all of the structurally different models leads to product lines that achieve good performance over all of the models and improve on each single-model product line in the worst-case.

A number of possibilities exist for further exploring the methodology presented here. Apart from the choice model, there exist many other sources of uncertainty. One such source of uncertainty is in the cost information; we may not precisely know the incremental marginal cost of each attribute. Another source of uncertainty is in the no-purchase behavior. In particular, we may not precisely know the competitive offerings that give rise to the no-purchase option; moreover, these competitive offerings may be decided in response to the product line that we offer. One could adapt the worst-case framework that we have described here to accommodate these considerations by populating the uncertainty set with models that correspond to different competitive response scenarios.

Chapter 4

Data-driven assortment optimization

4.1 Introduction

A ubiquitous element of business is that of making an assortment decision, which can be described most generally as follows. A firm offers a set of products (*an assortment*) to a group of individuals. The individuals possess preferences over the different products, which may vary from individual to individual, and proceed to select the product that they most prefer; the firm then garners some revenue from the choices of the individuals. The problem of *assortment optimization* is to decide what set of products, taken from a larger set of possible products, should be offered so as to maximize the firm's expected revenue when customers exercise their preferences.

Assortment optimization problems arise in many contexts:

- **Retail**. As a concrete example, consider a grocery store. A grocery store has many different product categories (e.g., coffee, soft drinks, breakfast cereals and so on). Within each category, the grocery store has many different individual products that could be stocked on its shelves (e.g., for coffee, the grocery store might have different roasts of different brands in different sizes). The grocery store must decide which ones to stock on its shelves so as to maximize its expected revenue per customer.

- **Online advertising**. In online advertising, part of a webpage is devoted to

displaying ads that are managed by an advertising platform (such as Google's AdSense system). The advertising platform has limited display space and must decide which ads to display so as to maximize the expected revenue (paid by the advertiser to the platform) per page view.

- **Social security**. As part of its social security system, a government may offer its citizens a number of funds that they can invest in for retirement. These funds may vary with respect to the type of securities, the industries represented in those securities and their overall risk-return performance. The government must then decide which funds to allow its citizens to invest in so as to ensure the maximum average return for its population while limiting the risk to the total investment of its citizens. A similar problem is faced by private employers who must decide on the assortment of 401(k) retirement savings plan to offer to their employees.

Such decisions have immense impact. In the retail domain, [48] describe how the grocery store Super Fresh stopped carrying certain dry grocery items in order to make room for fresh offerings, which lead to customers taking their business to other stores and ultimately, to the store declaring bankruptcy. In social security, [90] describe how the Swedish government in 2000 undertook an effort to privatize and reform the country's social security system. The government allowed private funds meeting certain criteria to be offered and encouraged its citizens to choose from among the resulting 456 different funds. The authors compare those who actively selected their portfolios with those who simply invested in a carefully designed default fund. The authors write:

> [Those] who selected portfolios for themselves selected a higher equity exposure, more active management, much more local concentration, and higher fees. ... Because of the decline in the market that followed the launch of this plan, investors did not do well for the first three years (from October 31, 2000, through October 31, 2003), but those who invested in the default fund suffered less. The default fund lost 29.9 percent in those

> three years, while the average portfolio of those participants who picked their funds actively lost 39.6 percent. ... Through July 2007 the default fund is up 21.5 percent while the average actively managed portfolio is up only 5.1 percent. [90]

The importance of assortment decisions is also underscored by the numerous commercial planning tools that are available or under development such as IBM's DemandTec® Assortment Optimization tool [61], JDA Assortment Optimization [64], Oracle Retail [72] and Celect's Choice Engine [31].

Assortment decisions in practice are difficult for a number of reasons:

1. **Choice modeling**. Most assortment decisions are affected by *demand substitution*. Each customer has preferences for the different products, and would ideally want to be able to choose the product from the universe of products that he prefers the most. However, if this product is not present in the assortment, he will select the product in the assortment that is most preferable to him. To model demand with substitution, there exists a wide variety of discrete choice models that may be used, such as the multinomial logit (MNL), nested logit (NL) and mixed multinomial logit (MMNL) models [10]. This poses a challenge because the decision maker is faced with the question of which model to use. If the model is too simple, it may underfit the available data. If it is too complex, it may overfit the available data, leading to inaccurate predictions.

2. **Tractability**. Although in some cases the underlying choice model leads to an assortment optimization problem that can be easily solved, typically the resulting problem is difficult from a computational complexity standpoint. There is also often an undesirable tradeoff between predictive accuracy and tractability: simple but inaccurate choice models lead to easy optimization problems, while complex choice models lead to intractable problems.

3. **Constraints**. In many assortment contexts, the firm may have various business rules that limit the possible assortments. A basic constraint is that of capacity;

for instance, a retailer may have limited shelf space, and an advertiser may have limited space on a webpage to fit ads. However, there may be many other constraints arising from the firm's internal rules that could affect assortment decisions. For example, a retailer may require that some products be offered together (e.g., if brand X dark roast coffee is offered, then brand Y dark roast coffee must also be offered), that only some number of products within a certain subclass of products is offered (e.g., of all dark roast coffee brands, offer at most three) or require at least some number of products within a given subset of products to be offered (e.g., offer at least two medium roast coffee products).

In this chapter, we propose a new approach for making assortment decisions that directly addresses the above challenges. The approach is based on modeling the choice behavior of the market through a generic ranking-based model of choice, and then using this ranking-based choice model to formulate and solve a mixed-integer optimization (MIO) problem that directly yields the optimal assortment decision. We make the following specific contributions:

1. We present a new approach for making assortment decisions based on modeling and solving the problem as an MIO problem. The optimization problem assumes a generic non-parametric choice model where the market is represented by a probability distribution over a finite, fixed collection of rankings over the products; such a model is able to capture any choice model based on random utility maximization, and can be readily formulated in an MIO framework. The optimization problem is efficient from a modeling perspective, as the number of binary variables scales with the number of products and the problem possesses desirable structure that facilitates its solution by standard solution techniques (specifically branch-and-bound). Using standard MIO modeling techniques, the problem also readily accommodates constraints.

2. We present an associated procedure for estimating the ranking-based choice model required by our model. The approach is based on solving a large-scale linear optimization (LO) problem that aims to minimize the ℓ_1 error between the

choice probabilities predicted by the model and the actual choice probabilities that are given by the data. We propose an efficient solution procedure based on column generation to find a finite collection of rankings and an associated probability distribution over them.

3. We demonstrate the effectiveness of our approach computationally:

   (a) We show that our assortment optimization model is practically efficient. Using commercial solvers, we show that we are able to solve very large instances of the basic assortment optimization problem with large numbers of products and large numbers of rankings comprising the underlying choice model *to full optimality*. Moreover, we show that in a large number of cases, the solution of the LO relaxation is integral, and when it is not, the relaxation gap is very small.

   (b) We show that constraints are easily modeled in our framework and have minimal impact on the practical efficiency of our model. To evaluate the benefit of our MIO framework, we adapt a local search procedure that delivers excellent practical performance on "simple" problems, namely unconstrained and cardinality constrained assortment problems, for more complex assortment optimization cases. We show that our MIO approach outperforms this local search procedure in these more complex cases.

   (c) We show that our estimation approach is tractable and yields accurate ranking-based models. To demonstrate this, we generate transaction data according to a known choice model, estimate our ranking-based model using a portion of the data and evaluate the accuracy of the estimated model on the remainder of the data. We show that our approach becomes more accurate with more data, is relatively resistant to overfitting and makes more accurate revenue predictions than a simple parametric model and a state-of-the-art non-parametric approach.

   (d) We show that combining our estimation and optimization approaches leads to effective decisions. To demonstrate this, we apply our estimation pro-

cedure to a set of training data, solve our MIO problem to obtain an assortment and then compare the *true* revenue of this assortment, i.e., under the same model that generated the data, to the best possible revenue under the same model. We show that our combined approach leads to revenues that are only a few percent from the optimal, that improve with more data and that are higher than revenues obtained from optimizing the aforementioned alternative parametric and non-parametric models.

The rest of this chapter is organized as follows. In Section 4.2, we review the relevant literature in choice modeling and assortment optimization. In Section 4.3, we present our modeling framework by describing the underlying ranking-based choice model and our MIO model for making assortment decisions given such a choice model. In Section 4.4, we present our estimation procedure for producing a ranking-based choice model from transaction data. In Section 4.5, we report on the results of our computational experiments. Lastly, in Section 4.6, we summarize our findings and propose directions for future research.

4.2 Literature review

Assortment optimization is a central topic in the operations management research literature; for a comprehensive overview of the topic, we refer the reader to the book chapter of [66]. We divide our discussion of how this work fits into the existing research landscape according to two areas: choice modeling and optimization.

Choice modeling. The primary prerequisite for assortment optimization is a choice model that specifies, for a given set of products, how frequently customers will select each product in the set. There exists a variety of choice models one could apply; the reader is referred to the excellent literature review of [44] for a concise overview and the books of [10] and [93] for a more detailed treatment. One of the most basic parametric choice models is the multinomial logit model (MNL) model. Although the MNL model is widely used, it exhibits certain undesirable properties, such as the

independence of irrelevant alternatives (IIA) property [10]. As a result, more complex parametric models such as the mixture of multinomial logits (MMNL) model and the nested logit (NL) model have been proposed.

While parametric models have proven to be accurate in many applications, it is often not easy to decide on which parametric model to use. A simple model such as the MNL model may underfit the data and ignore important choice phenomena in the data, potentially leading to poor predictions of choice frequencies and suboptimal assortment decisions. At the same time, a more complex model such as the NL model may overfit the data and also lead to poor predictions and suboptimal assortment decisions.

As an alternative to parametric models, a number of generic choice models have been proposed that make minimal structural assumptions and that are capable of representing a wide variety of choice models. In particular, [44] proposed a general model of choice where one represents choice behavior by a probability distribution over all of the possible rankings of the products. For a given assortment, [44] then propose predicting the expected revenue of the assortment by computing the worst-case expected revenue, where the worst-case is taken over all probability distributions that are consistent with the available transaction data. They show, using both synthetic and real data, that their revenue predictions are more accurate than those produced by parametric models such as MNL and MMNL. Another general model that has recently been proposed is the Markov chain model of customer choice [25]. In this model, products are modeled as states in a Markov chain and substitution behavior is modeled by transitions in the same Markov chain; the choice probabilities for a given offer set can then be computed as the absorption probabilities of the products in the offer set. They show that such a model provides a good approximation to any choice model based on random utility maximization. Note, however, that the Markov chain choice model as presented in [25] can only be estimated when one has historical data corresponding to a specific set of $n + 1$ assortments, where n is the number of products; the estimation of the model when one has an arbitrary collection of historical assortments remains an open problem.

The assortment optimization approach that we present in this chapter builds upon the framework of [44] in that we also consider a probability distribution over rankings, but there is a critical difference. As mentioned above, [44] is concerned with predicting revenues by evaluating the worst-case revenue over all probability distributions that reconcile the available data. In contrast, our approach does not consider the worst case. Rather, we fix a small set of rankings over the products and a probability distribution over this small set of rankings. This is important for three reasons. The first reason is that revenue predictions under our framework are considerably simpler to compute than in the worst-case approach. In our framework, for a given set of products, one simply computes the most preferred product under each ranking and sums the corresponding revenues weighted by the probability distribution. On the other hand, evaluating the worst-case revenue when the support of the probability distribution is the set of *all* possible ranking is more challenging, as one has to solve an LO problem with an intractable number of decision variables (one per ranking of products; for n products and a single no-purchase alternative, this results in $(n+1)!$ decision variables) [44]. The second reason, which is related to the first, is that in our framework, the corresponding assortment optimization problem can be compactly formulated as an MIO whose size scales gracefully in the number of products and the number of rankings. In contrast, in the worst-case framework, it is not clear how one may formulate the problem of optimizing the worst-case revenue as an efficiently solvable mathematical optimization formulation. The most promising proposal for solving this problem is the ADXOpt algorithm, proposed by [62], which is a local search algorithm and cannot guarantee global optimality. The third and final reason is that there are substantial differences in predictive accuracy. As we will show later, fixing a set of rankings and a probability distribution over them leads to more accurate predictions of revenue than the worst-case approach (Section 4.5.4), and ultimately leads to better decisions (Section 4.5.6).

This conceptual difference between our work and the framework of [44] leads to the question of estimation: given some transaction data, what is a set of rankings and an associated probability distribution that would give rise to this data? In response, we

propose a simple estimation procedure for identifying such a probability distribution and a set of rankings. Our procedure is based on solving an ℓ_1 approximation problem using column generation. To the best of our knowledge, the only other procedure for fitting this kind of choice model – a probability distribution over the set of rankings – is the approach of [95]. There are some similarities between our procedure and that of [95]; most notably, both we and [95] propose a column generation step that involves solving essentially the same subproblem. However, the main difference between our proposal and that of [95] is that we approach the model from an ℓ_1 approximation perspective whereas [95] use maximum likelihood estimation. This turns out to be important, because one can find optimal solutions to our problem that are basic feasible solutions, where only a small number of rankings have non-zero probability; this limits the complexity of the model that our procedure identifies and guards against overfitting. On the other hand, in the framework of [95], one must perform some kind of model selection to prevent overfitting.

Optimization. Alongside the body of research devoted specifically to modeling choice, there is a rich literature that considers assortment optimization under these models – some examples include the MNL model [89], the robust MNL model [82], the NL model [37, 69], the MMNL model [30, 81] and the Markov chain choice model [25, 46]. There is also research that considers assortment optimization under constraints such as cardinality constraints [80, 49], capacity constraints [41, 49, 47] and other constraints such as precedence and quality consistency [36].

The majority of prior research on assortment optimization fits what can be summarized as a "fix-then-exploit" approach: one *fixes* a given parametric choice model and then *exploits* the structure of the resulting optimization problem to develop exact solution procedures, if possible, or otherwise heuristics or approximation algorithms. On the other hand, the approach we propose and advocate for is to estimate a ranking-based model of choice from the data and to formulate the assortment optimization problem as an MIO problem. We believe that our approach carries a number of advantages. First, in our approach, there is no "problem-specific" effort that is required:

MIO problems can always be solved exactly via branch-and-bound to obtain a solution that is provably optimal or has a guarantee on its suboptimality (if the process is terminated early). Second, MIO as a methodology is highly flexible, in that (1) we can easily model a variety of business rules as linear constraints in the MIO problem, and (2) we can then simply "declare" these constraints to the solver, and let the solver do the work of accounting for them. In contrast, constraints are highly problematic in the fix-then-exploit approach, in that they necessitate entirely different algorithms relative to the unconstrained case. For example, the unconstrained MNL problem can be solved by enumerating the n revenue-ordered subsets [89]; however, with capacity constraints, the problem becomes *NP-Hard* and one must consider more complex approximation algorithms [41].

Of special note is the ADXOpt algorithm, proposed by [62]. ADXOpt is a local search procedure that starts with the empty set and in each iteration, moves to a new assortment that most improves the expected revenue by considering additions of products outside of the assortment, deletions of products inside of the assortment and exchanges of products inside the assortment with ones outside. The procedure is general in that the only prerequisite for it is a function that maps an assortment to its expected revenue. Moreover, the procedure identifies the exact optimal assortment for certain choice models and delivers excellent performance in numerical experiments with other choice models (such as the model of [44]). Although this approach is simple, it is not designed to handle complex constraints, nor is it clear how to extend it to handle them. To demonstrate the impact of this, in Section 4.5.2 we consider a reasonable modification to ADXOpt to accommodate constrained problems – namely, to only consider local moves that maintain feasibility – and we show that for moderately constrained problems, ADXOpt can be significantly suboptimal compared to our MIO approach.

Outside of operations management, assortment optimization is connected to the problem of product line design (PLD), which is a central problem in marketing science. In the PLD problem, a firm seeks to introduce a product that has certain attributes to a market of heterogeneous customers. The firm must decide on which versions

of the product, as defined by the attributes, to offer so as to maximize the revenue garnered from the customer's choices. This problem has been studied extensively and there exist many IO formulations of it (see the literature review in Chapter 3). The MIO formulation that we describe is most similar to the formulation of the PLD problem used in [9], which is described in the electronic companion of that paper. Our formulation refines the formulation of [9] by modeling a large number of variables as continuous rather than binary by exploiting a property of the constraints. However, the scales of the problems that we are solving are different and as such, our research generates substantially different insights. The paper of [9] solved the PLD for a real problem involving 3584 candidate products and over 300 preference rankings of these products; even after implementing various sophisticated MIO enhancements, such as Lagrangian relaxation and valid inequalities, the problem required *over a week* to solve to full optimality and as such, it is not practical for product line design problems. However, in our computational experiments, the corresponding assortment optimization problems we are faced with are considerably smaller and as such, can be solved to full optimality within *seconds* and *without* additional enhancements of the kind used in [9].

4.3 Assortment optimization

In this section, we describe our assortment optimization method. We begin in Section 4.3.1 by describing our ranking-based choice model; then, in Section 4.3.2 we present our MIO formulation for finding the optimal assortment under this choice model.

4.3.1 Choice model

We begin by defining the non-parametric choice model that describes the choice behavior of the market. The model we will consider is conceptually similar to the model used in [44]; the reader is referred to [44] for further details.

We assume that there are n products, indexed from 1 to n. We use the index 0 to

denote the no-purchase alternative (the possibility that the customer does not purchase any of the products that we offer). Together, we refer to the set $\{0, 1, 2, \ldots, n\}$ – the set of products together with the no-purchase alternative – as the *options* that are available. We assume that a customer will select exactly one of the available options. We also assume that the no-purchase alternative 0 is always available to the customer.

We assume that we have K rankings or permutations $\sigma^1, \ldots, \sigma^K$ over the options $\{0, 1, 2, \ldots, n\}$. Each permutation $\sigma^k : \{0, 1, 2, \ldots, n\} \to \{0, 1, 2, \ldots, n\}$ is a bijection that assigns each option to a rank in $\{0, 1, 2, \ldots, n\}$. The value $\sigma^k(i)$ indicates the rank of option i; $\sigma^k(i) < \sigma^k(j)$ indicates that i is more preferred to j under the permutation σ^k. We can think of each σ^k as a "mode" or "type" of choice behavior that a customer may follow. We assume that given a set of products $S \subseteq \{1, 2, \ldots, n\}$, a customer that follows the permutation σ^k will select the option i from the set $S \cup \{0\}$ with the lowest rank, i.e., the option $\arg\min_{i \in S \cup \{0\}} \sigma^k(i)$.

We use λ^k to denote the probability that a random customer makes a choice according to the permutation σ^k; we use $\boldsymbol{\lambda}$ to denote the probability mass function (PMF) over the set of permutations $\{\sigma^1, \ldots, \sigma^K\}$. For a given assortment of products $S \subseteq \{1, 2, \ldots, n\}$, the probability $\mathbb{P}(i \,|\, S)$ that a random customer selects option $i \in \{0, 1, 2, \ldots, n\}$ given that the available set of products was S is given by

$$\mathbb{P}(i \,|\, S) = \sum_{k=1}^{K} \lambda^k \cdot \mathbb{I}\{i = \arg \min_{i' \in S \cup \{0\}} \sigma^k(i')\},$$

where $\mathbb{I}\{\cdot\}$ is the indicator function ($\mathbb{I}\{A\}$ is 1 if A is true and 0 otherwise).

Let r_i be the revenue of option i; we assume that the revenue r_0 of the no-purchase alternative is exactly zero. Then the expected revenue from offering the set of products

S is denoted by $R(S)$ and is given by

$$\begin{aligned} R(S) &= \sum_{i \in S} r_i \cdot \mathbb{P}(i \,|\, S) \\ &= \sum_{i \in S} r_i \cdot \left(\sum_{k=1}^{K} \lambda^k \cdot \mathbb{I}\{i = \arg \min_{i' \in S \cup \{0\}} \sigma^k(i')\} \right). \end{aligned}$$

4.3.2 Mixed-integer optimization model

Having defined the non-parametric choice model we will be using, we now turn our attention to how to make assortment decisions. The problem we wish to solve is to find the set of products S^* that maximizes the expected revenue:

$$S^* = \arg \max_{S \subseteq \{1,\ldots,n\}} R(S).$$

We will formulate the problem as an MIO problem. For each product $i \in \{1, \ldots, n\}$, let x_i be a binary decision variable that is 1 if product i is included in the assortment, and 0 otherwise. For each option $i \in \{0, 1, \ldots, n\}$, let y_i^k be a decision variable that is 1 if option i is chosen under the kth permutation, and 0 otherwise.

The problem, in its most basic form, can then be formulated as follows.

$$\begin{aligned}
\underset{\mathbf{x},\mathbf{y}}{\text{maximize}} \quad & \sum_{k=1}^{K} \sum_{i=1}^{n} r_i \cdot \lambda^k \cdot y_i^k && \text{(4.1a)} \\
\text{subject to} \quad & \sum_{i=0}^{n} y_i^k = 1, \quad \forall\, k \in \{1, \ldots, K\}, && \text{(4.1b)} \\
& y_i^k \leq x_i, \quad \forall\, k \in \{1, \ldots, K\},\ i \in \{1, \ldots, n\}, && \text{(4.1c)} \\
& \sum_{j : \sigma^k(j) > \sigma^k(i)} y_j^k \leq 1 - x_i, \quad \forall\, k \in \{1, \ldots, K\},\ i \in \{1, \ldots, n\}, && \text{(4.1d)} \\
& \sum_{j : \sigma^k(j) > \sigma^k(0)} y_j^k = 0, \quad \forall\, k \in \{1, \ldots, K\}, && \text{(4.1e)} \\
& x_i \in \{0, 1\}, \quad \forall\, i \in \{1, \ldots, n\}, && \text{(4.1f)} \\
& y_i^k \geq 0, \quad \forall\, k \in \{1, \ldots, K\},\ i \in \{0, 1, \ldots, n\}. && \text{(4.1g)}
\end{aligned}$$

In order of appearance, the constraints have the following meaning. Constraint (4.1b) ensures that under each ranking, exactly one choice is made. Constraint (4.1c) ensures that under ranking k, product i can only be chosen if product i is included in the assortment. Constraint (4.1d) ensures that, if product i is included in the assortment, then none of the options that are less preferred to i under ranking σ^k may be chosen under ranking k (the y_j^k variables for all j that are less preferred are forced to zero). Constraint (4.1e) is a similar constraint, but pertaining to the no-purchase option: those options that are less preferred to the no-purchase option 0 may not be selected and are forced to zero. The penultimate constraint ensures that the x_i's are binary. The last constraint ensures that each y_i^k may be nonnegative.

It is worth commenting on several important aspects of this formulation. First, note that the formulation does not require the y_i^k variables to be binary. This is because for fixed binary values of x_i, constraints (4.1b)–(4.1e) ensure that the y_i^k values are forced to the correct binary values. Thus, as a result, the formulation has only n binary variables (the x_i variables).

Second, note that the structure of the formulation allows for relatively efficient optimization via branch-and-bound. In particular, suppose that at some point in the branch-and-bound tree, we branch on variable x_i and set it to 1. Then, constraint (4.1d) forces y_j^k for all j that are less preferred to i to be zero. It is plausible that for many rankings k, there may be many options j that are less preferred to i; as a result, branching up on the variable x_i changes the revenue significantly, allowing for other nodes in the branch-and-bound tree to be pruned.

Finally, note that although we have formulated the problem in a rather basic way, we can expand it to incorporate different kinds of business requirements by adding constraints to the formulation. We provide some examples below.

1. **Lower and upper bounds on the size of the assortment.** If the set of possible products is large, the firm may not want to offer too many products; similarly, the firm may also want to avoid situations where it offers too few products. If U and L are upper and lower bounds on the number of products

in the assortment, then this requirement can be modeled as follows:

$$L \leq \sum_{i=1}^{n} x_i \leq U.$$

2. **Subset constraints.** Similarly, the firm may have a requirement that out of a subset $S \subseteq \{1, \ldots, n\}$, of the products, at most U_S products may be included and at least L_S products must be included. This is also easily modeled; we have

$$L_S \leq \sum_{i \in S} x_i \leq U_S.$$

3. **Precedence constraints.** The firm may require that if a product i is included, then a product i' must be included. This is easily modeled as

$$x_i \leq x_{i'}.$$

4.4 Choice model estimation

In Section 4.3.1, we introduced a choice model that is defined by a set of K permutations $\sigma^1, \ldots, \sigma^K$ and a probability distribution $\boldsymbol{\lambda} = (\lambda^1, \ldots, \lambda^K)$ over this set. A natural question to ask is: where do the permutations and the probability distribution come from? One answer to this question is to estimate this model directly from transaction data, which we now describe. The type of transaction data that we will assume is available will be conceptually similar to the data that is assumed in [44]; we refer the reader to that paper for additional detail.

We assume that from previous operations, we have data corresponding to M assortments of products, $S_1, \ldots, S_M \subseteq \{1, 2, \ldots, n\}$. For each m and each option i in $S_m \cup \{0\}$, we assume that we have $v_{i,m}$ which is the probability with which customers selected option i when the assortment S_m was offered to them. We assume that $v_{i,m}$ is fully accurate and noiseless, i.e., $v_{i,m}$ is exactly equal to $\mathbb{P}(i \,|\, S_m)$ for each m and each $i \in S_m \cup \{0\}$. We use $\mathbf{v}$ to denote the vector of $v_{i,m}$ values. We will also use P

to denote the number of (i, m) pairs; $\mathbf{v}$ is then a vector of dimension P.

Let us assume, for a moment, that we consider the entire set of $(n+1)!$ permutations on the total set of options $\{0, 1, 2, \ldots, n\}$; that is, we set $K = (n+1)!$ and allow the collection $\sigma^1, \ldots, \sigma^K$ to correspond to all of the possible permutations of the options. Then, as in [44], we can define the matrix $\mathbf{A}$ as the matrix of $A^k_{i,m}$ values, where

$$A^k_{i,m} = \begin{cases} 1, & \text{if } i = \arg\min_{j \in S_m \cup \{0\}} \sigma^k(j), \\ 0, & \text{otherwise.} \end{cases}$$

In words, $A^k_{i,m}$ is 1 if under permutation k, the customer would have chosen option i from the set of options $S_m \cup \{0\}$, and is 0 otherwise. The matrix $\mathbf{A}$, vector $\mathbf{v}$ and probability mass function $\boldsymbol{\lambda}$ are related in that $\boldsymbol{\lambda}$ must satisfy the linear system of equations

$$\mathbf{A}\boldsymbol{\lambda} = \mathbf{v}.$$

Therefore, one way that we may identify $\boldsymbol{\lambda}$ is to solve a problem where we minimize the ℓ_1 error between $\mathbf{A}\boldsymbol{\lambda}$, which is the vector of predicted choice probabilities for the options in the assortments $S_1, \ldots, S_M$, and $\mathbf{v}$, which is the vector of actual choice probabilities for those same options. Mathematically, this problem can be stated as follows:

$$\begin{aligned} \underset{\boldsymbol{\lambda}}{\text{minimize}} \quad & \|\mathbf{A}\boldsymbol{\lambda} - \mathbf{v}\|_1 && \text{(4.2a)} \\ \text{subject to} \quad & \mathbf{1}^T\boldsymbol{\lambda} = 1, && \text{(4.2b)} \\ & \boldsymbol{\lambda} \geq \mathbf{0}. && \text{(4.2c)} \end{aligned}$$

Through standard LO modeling techniques (see, e.g., [23]), this problem can be reformulated into the following LO problem:

$$\begin{aligned} \underset{\boldsymbol{\lambda}, \boldsymbol{\epsilon}^+, \boldsymbol{\epsilon}^-}{\text{minimize}} \quad & \mathbf{1}^T\boldsymbol{\epsilon}^+ + \mathbf{1}^T\boldsymbol{\epsilon}^- && \text{(4.3a)} \\ \text{subject to} \quad & \mathbf{A}\boldsymbol{\lambda} + \boldsymbol{\epsilon}^+ - \boldsymbol{\epsilon}^- = \mathbf{v}, && \text{(4.3b)} \\ & \mathbf{1}^T\boldsymbol{\lambda} = 1, && \text{(4.3c)} \end{aligned}$$

$$\boldsymbol{\lambda}, \boldsymbol{\epsilon}^+, \boldsymbol{\epsilon}^- \geq \mathbf{0}. \tag{4.3d}$$

At first glance, the problem that we have arrived at may not appear to be useful. In particular, the number of λ^k variables is $(n+1)!$ – for even the smaller values of n that may occur in practice, the resulting number of variables is too large for the problem to be directly solved by off-the-shelf solvers. The value of modeling the estimation problem in this way is that we have cast it as a large-scale LO problem where the columns of the λ^k variables follow a specific combinatorial structure. Thus, we are able to apply column generation to solve the problem efficiently.

At a high level, our column generation procedure operates as follows. At each iteration, we maintain a collection of permutations $\sigma^1, \ldots, \sigma^K$. We refer to problem (4.3) with the full set of $(n+1)!$ permutations as the *master* problem, and with the restricted collection $\sigma^1, \ldots, \sigma^K$ of permutations as the *restricted master* problem. We begin the procedure with no permutations. We solve the corresponding restricted master. From the resulting optimal solution, we solve the subproblem to identify the new permutation σ to add to $\sigma^1, \ldots, \sigma^K$ with the lowest reduced cost. If the optimal value of the subproblem is negative, we add the corresponding permutation to $\sigma^1, \ldots, \sigma^K$, update K and go back and solve the new restricted master problem again; otherwise, if the optimal value of the subproblem is nonnegative, we terminate the procedure.

We now discuss the structure of the subproblem. Let $\boldsymbol{\alpha}$ and ν be the dual variables corresponding to constraints (4.3b) and (4.3c), respectively. We seek to find the permutation σ and the corresponding column $\mathbf{a}$ of the matrix $\mathbf{A}$ such that the reduced cost of the corresponding λ is as negative as possible. Recall that for a permutation σ, the column $\mathbf{a}$ is defined component-wise as $a_{i,m} = 1$ if under permutation σ, option i is selected from the set of options $S_m \cup \{0\}$ and 0 otherwise. To model the permutation σ, we introduce a binary variable z_{ij} for each distinct i and j in $\{0, 1, 2, \ldots, n\}$ that takes the value 1 if under σ, i is preferred to j ($\sigma(i) < \sigma(j)$) and 0 otherwise. The

subproblem can then formulated as an integer optimization (IO) problem:

$$\underset{\mathbf{z},\mathbf{a}}{\text{minimize}} \quad 0 - \boldsymbol{\alpha}^T \mathbf{a} - \nu \tag{4.4a}$$

$$\text{subject to} \quad \sum_{i \in S_m \cup \{0\}} a_{i,m} = 1, \quad \forall\, m \in \{1, \ldots, M\}, \tag{4.4b}$$

$$a_{i,m} \leq z_{ij}, \quad \forall\, m \in \{1, \ldots, M\},\ i, j \in S_m \cup \{0\},\ i \neq j, \tag{4.4c}$$

$$z_{ij} + z_{ji} = 1, \quad \forall\, i, j \in \{0, 1, 2, \ldots, n\},\ i \neq j, \tag{4.4d}$$

$$z_{ij} + z_{jk} - 1 \leq z_{ik}, \quad \forall\, i, j, k \in \{0, 1, 2, \ldots, n\}, \\ i \neq j,\ i \neq k,\ j \neq k, \tag{4.4e}$$

$$z_{ij} \in \{0, 1\}, \quad \forall\, i, j \in \{0, 1, 2, \ldots, n\},\ i \neq j, \tag{4.4f}$$

$$a_{i,m} \in \{0, 1\}, \quad \forall\, m \in \{1, \ldots, M\},\ i \in S_m \cup \{0\}. \tag{4.4g}$$

In order of appearance, the constraints have the following meaning. Constraint (4.4b) ensures that in each portion of $\mathbf{a}$ corresponding to a single assortment S^m, exactly one of the entries is one (i.e., the ranking must select one of the options in $S^m \cup \{0\}$). Constraint (4.4c) links the values in the column $\mathbf{a}$ with the values of the permutation; in particular, if $a_{i,m} = 1$, then it must be that $z_{ij} = 1$ for every other j in $\mathcal{S} \cup \{0\}$. Constraint (4.4d) represents non-reflexivity: either i is ranked lower than j or j is ranked lower than i. Constraint (4.4e) represents transitivity: for any three distinct options i, j and k, it must be that if i is ranked lower than j and j is ranked lower than k, then i is ranked lower than k. The objective function corresponds to the reduced cost of the permutation encoded by the $\mathbf{z}$ variables.

The full column generation procedure is presented as Algorithm 3.

We conclude our discussion of our estimation procedure by discussing two practical modifications that we will employ in our numerical experiments in Section 4.5. First of all, the current stopping criterion ensures that we solve the problem to full optimality; this happens when the objective $\|\mathbf{A}\boldsymbol{\lambda} - \mathbf{v}\|_1 = 0$, or equivalently, $\boldsymbol{\lambda}$ exactly solves $\mathbf{A}\boldsymbol{\lambda} = \mathbf{v}$. Alternatively, we may be satisfied with an approximate solution to this system of equations. To obtain an approximate solution, we may consider terminating

Algorithm 3 Column generation algorithm.

Require: Choice probability vector $\mathbf{v}$, training assortments $S_1, \ldots, S_M$.
  Initialize K to 0.
  Set $\mathbf{A}$ to be a $P \times 0$ (empty) matrix.
  Solve restricted master problem (4.3) with $\mathbf{A}$ to obtain dual variable values $\boldsymbol{\alpha}$ and ν.
  Solve subproblem (4.4) with $\boldsymbol{\alpha}$, ν to obtain $\mathbf{z}$, $\mathbf{a}$.
  while $-\boldsymbol{\alpha}^T\mathbf{a} - \nu < 0$ **do**
    Update $K \leftarrow K + 1$.
    Set σ^K as $\sigma^K(i) = \sum_{\substack{j=0 \\ j \neq i}}^{n} z_{ji}$.
    Set $\mathbf{A} \leftarrow [\mathbf{A}\ \mathbf{a}]$.
    Solve restricted master problem (4.3) with $\mathbf{A}$ to obtain primal variable values $\lambda^1, \ldots, \lambda^K$, dual variable values $\boldsymbol{\alpha}$ and ν.
    Solve subproblem (4.4) with $\boldsymbol{\alpha}$, ν to obtain $\mathbf{z}$, $\mathbf{a}$.
  end while
  return $\sigma^1, \ldots, \sigma^K$ and $\lambda^1, \ldots, \lambda^K$.

when the objective value of the restricted master problem is close enough to zero, i.e., when $\|\mathbf{A}\boldsymbol{\lambda} - \mathbf{v}\|_1 \leq P \cdot \epsilon$ for some $\epsilon > 0$. Observe that the quantity $\|\mathbf{A}\boldsymbol{\lambda} - \mathbf{v}\|_1 / P$ can be interpreted as the mean absolute error (MAE) of the current solution $\lambda^1, \ldots, \lambda^K$, $\sigma^1, \ldots, \sigma^K$ on the training set; thus, ϵ represents the training set MAE that we wish to achieve.

Second of all, note that although the subproblem (4.4) is an IO problem, it is not necessary to solve it as such. In particular, any solution $(\mathbf{z}, \mathbf{a})$ that is feasible for problem (4.4) and achieves a negative objective value corresponds to a permutation σ whose λ variable may enter the basis. Thus, rather than solving subproblem (4.4) exactly, we may opt to solve it approximately via a local search procedure. We consider a local search procedure that operates as follows. Starting from some initial (randomly chosen) permutation σ, we consider all neighboring permutations σ' obtained by taking σ and swapping the rankings of any two distinct options. We evaluate the reduced cost of each such σ'; if no neighboring σ' improves on the reduced cost of σ, we terminate with σ as the locally optimal solution. Otherwise, we move to the σ' that most improves the reduced cost of σ and repeat the procedure at this new permutation. If the locally optimal permutation does not have negative reduced cost, we can repeat the search starting at a new random permutation; we continue doing

so until we find a locally optimal permutation with negative reduced cost or we have reached the maximum number of repetitions. Our preliminary experimentation with using this local search procedure within the column generation procedure suggested that it could find an approximate solution (satisfying $\|\mathbf{A}\boldsymbol{\lambda}-\mathbf{v}\|_1 \leq P\cdot\epsilon$) more rapidly than by solving problem (4.4) directly as an IO problem.

4.5 Computational results

In this section, we report on the results of our computational experiments. Our insights are as follows:

- In Section 4.5.1, we show that our MIO model is practically tractable. We show that it can be solved rapidly for large instances (large numbers of products and rankings), the LO relaxation provides a good approximation of the integer optimal value and that in a large proportion of instances, the LO is in fact integral.

- In Section 4.5.2, we show that constraints have a negligible impact on how efficiently problem (4.1) can be solved to full optimality. At the same time, we show that the ADXOpt local search procedure of [62], which achieves strong performance in unconstrained and cardinality constrained problems, can be significantly suboptimal in the presence of complex constraints.

- In Section 4.5.3, we show that our estimation procedure can quickly obtain ranking-based models that yield accurate out-of-sample predictions of choice probabilities. We show that the procedure is relatively resistant to overfitting and learns more accurate models with more training data.

- In Section 4.5.4, we compare predictions of expected revenues from our approach to revenue predictions from (1) an MNL model fitted to the same data and (2) the worst-case approach of [44]. Although the MNL model is more accurate in some instances, it can be significantly less accurate than our approach when the

underlying model is not an MNL model; moreover, even with additional data, it is not able to learn. The worst-case predictions using the approach of [44] are in general much less accurate than those produced by our approach.

- In Section 4.5.5, we show that by combining our estimation and optimization procedures, we find assortments that achieve expected revenues within a few percent of the (unknown) true optimal revenue. As the amount of data increases, this optimality gap shrinks. Analogously to the lack of overfitting in prediction, we show that our combined estimation and optimization method is resistant to overfitting in optimization: as the number of column generation iterations increases, the optimality gap does not deteriorate but in fact improves.

- In Section 4.5.6, we compare our assortments against the fitted MNL assortments and the worst-case-optimal assortments (as determined by ADXOpt). We find that our approach is better over a wider range of models than the fitted MNL approach. We also find that our approach significantly outperforms the worst-case optimal approach.

We implemented our experiments in the Julia technical computing language [24]. All mathematical optimization problems were modeled using the JuMP package for Julia [70]. All linear and mixed-integer linear optimization problems were solved using Gurobi 5.60 [56] and all nonlinear optimization problems were solved using IPOPT [96].

4.5.1 Tractability of assortment optimization model

To test the tractability of the assortment optimization formulation (4.1), we consider the following experiment. For fixed values of the number of products n and the number of permutations K, we randomly generate 100 instances, where we uniformly at random generate the set of rankings $\sigma^1, \ldots, \sigma^K$ from the set of all possible permutations, the revenue r_i of each product i from the set $\{1, \ldots, 100\}$, and the probability distribution $\boldsymbol{\lambda}$ from the $(K-1)$-dimensional unit simplex. For each of

these 100 instances, we solve both the LO relaxation and the actual MIO problem itself. We then record the average time to solve the MIO, the average time to solve the LO relaxation, the average relaxation gap (where the relaxation gap is defined as $100\% \times (Z_{LO} - Z_{MIO})/Z_{LO}$, where Z_{LO} and Z_{MIO} are the optimal values of the LO relaxation and the true MIO formulation, respectively) and the percentage of instances where the LO solution turned out to be integral.

Table 4.1 reports on the above metrics for different values of n in $\{10, 20, 30, 40\}$ and values of K in $\{10, 100, 200, 500, 1000\}$. From Table 4.1, we can see that the MIO formulation is very tractable; in the largest collection of instances ($n = 40$, $K = 1000$), problem (4.1) can be solved in approximately 10 seconds on average. Moreover, the formulation is efficient, in the sense that the relaxation is a good approximation of the true integer formulation; for each value of n and K, the average gap of the LO relaxation and the true MIO model is no more than 1%, and in a large number of cases (more than 20% for each combination of n and K) the LO solution is integral.

4.5.2 Constrained assortment optimization

We next show the value of using our optimization model to accommodate constraints on the assortment. To do this, we consider the collection of instances from Section 4.5.1 corresponding to $n = 30$ products and $K = 100$ permutations. For each instance, we consider the corresponding assortment optimization problem with a randomly generated set of constraints. We solve the constrained assortment optimization problem exactly using our MIO model (4.1) and using the ADXOpt local search heuristic proposed by [62]. We adapt the ADXOpt local search heuristic to the constrained setting by only allowing additions, deletions or exchanges that ensure that the assortment will remain feasible at each iteration. We limit ADXOpt to one removal of each product.

We consider several different types of constraint sets:

1. **No constraints**. The corresponding problem is the unconstrained problem.

2. **Maximum subset**. A maximum subset constraint set is parametrized by an

n	K	MIO Time	Rlx. Gap (%)	% Integral	LO Time (s)
10	10	0.0	0.30	81.0	0.0
10	100	0.0	0.41	69.0	0.0
10	200	0.1	0.44	65.0	0.0
10	500	0.2	0.42	66.0	0.1
10	1000	0.4	0.44	63.0	0.2
20	10	0.0	0.11	85.0	0.0
20	100	0.1	0.60	55.0	0.0
20	200	0.2	0.48	46.0	0.1
20	500	0.8	0.62	42.0	0.2
20	1000	1.8	0.58	50.0	0.7
30	10	0.0	0.19	82.0	0.0
30	100	0.2	0.52	46.0	0.1
30	200	0.5	0.56	41.0	0.1
30	500	1.9	0.69	36.0	0.5
30	1000	5.2	0.88	29.0	1.6
40	10	0.0	0.11	75.0	0.0
40	100	0.4	0.45	43.0	0.3
40	200	1.1	0.69	33.0	0.2
40	500	3.6	0.89	23.0	0.8
40	1000	10.0	0.71	19.0	2.7

Table 4.1: Results of tractability experiment.

integer C that specifies the number of constraints, an integer B that specifies the number of products that participate in each constraint and an integer U which specifies the maximum number of products we may select from the B products. It has the form

$$\sum_{i=B(c-1)+1}^{Bc} x_{(i)} \leq U, \quad \forall\, c \in \{1, \ldots, C\},$$

where (j) is the product with the jth highest marginal revenue. For example, with $C = 2$, $B = 3$ and $U = 2$, the constraint set is

$$x_{(1)} + x_{(2)} + x_{(3)} \leq 2,$$
$$x_{(4)} + x_{(5)} + x_{(6)} \leq 2,$$

which means that from among the top three products in marginal revenue, we may include at most two products, and from the next three products, we also may include at most two products. Such a constraint set could model a requirement that the assortment be diverse and not be biased towards certain groups of products.

3. **Precedence type 1**. A type 1 precedence constraint set is parametrized by an integer C that specifies the number of constraints and an integer B that specifies the number of products that participate in each constraint. It has the form

$$\sum_{j=B(c-1)+1}^{Bc} x_{(j)} \leq \sum_{j=Bc+1}^{B(c+1)} x_{(j)}, \quad \forall\, c \in \{1, \ldots, C\},$$

where, as for the maximum subset constraint set, (j) indicates the product with the jth largest marginal revenue. For example, for $C = 2$ and $B = 3$, the constraint set is

$$x_{(1)} + x_{(2)} + x_{(3)} \leq x_{(4)} + x_{(5)} + x_{(6)},$$
$$x_{(4)} + x_{(5)} + x_{(6)} \leq x_{(7)} + x_{(8)} + x_{(9)},$$

which means that the number of products we select from the top three products by marginal revenues (products (1) through (3)) cannot exceed the number of products we select from the next three products by marginal revenue (products (4) through (6)), which itself cannot exceed the number of products we select from the next three products by marginal revenue (products (7) through (9)). The constraint represents a form of precedence because including a product with a high marginal revenue mandates the inclusion of products with lower marginal revenues.

As with the maximum subset constraint, such a constraint could represent a business requirement that the firm must offer a diverse collection of products, and not offer just those products with the highest marginal revenue.

4. **Precedence type 2.** A type 2 precedence constraint set is parametrized by an integer C that specifies the number of constraints and an integer B that specifies the number of products that participate in each constraint. It has the form

$$\sum_{j \in J_c} x_j \leq (B-1) x_{j_c^*}, \quad \forall\, c \in \{1, \ldots, C\},$$

where for each c, J_c is a subset of $\{1, \ldots, n\}$ of size $B-1$, and j_c^* is a distinct product (i.e., not one of the products in J_c). In the experiments we will report on below, we randomly choose the set of products $J_c \cup \{j_c^*\}$ from $\{1, \ldots, n\}$ for each instance and each of the C constraints.

In words, this constraint requires that if the assortment includes a product in J_c, then it must include the "main" product j_c^*. This type of constraint could represent a requirement that the firm offer certain items together. For example, the product j_c^* could correspond to the "main" version of a product (e.g., regular Coca Cola) and the products in J_c could correspond to alternative versions of this product (e.g., Diet Coke and Coke Zero) that can only be offered if the main product is offered.

Note that for all of these constraint sets, the empty assortment (in terms of the MIO

formulation, this is represented by $x_i = 0$ for all $i \in \{1, \ldots, n\}$) is feasible and is still be used as the initial solution for ADXOpt, as originally described in [62].

For each instance and each such constraint set, we add the constraint set to problem (4.1) and solve it to obtain the optimal value Z^*_{MIO}. We also apply ADXOpt to solve the constrained problem with the modifications described above; we let the Z^*_{ADXOpt} denote the expected revenue of the ADXOpt assortment. For each instance and each constraint set, we compute the gap as $100\% \times (Z^*_{MIO} - Z^*_{ADXOpt})/Z^*_{MIO}$. We also record the time required to solve each instance under each constraint set for both the MIO and the ADXOpt approaches.

Table 4.2 reports the average gap of each constraint set, where the average is taken over the 100 instances, as well as the average time to solve the problem using our MIO approach and the ADXOpt approach. We can see that, in the absence of constraints, ADXOpt performs very well; the average optimality gap, over the 100 instances, is 0.07% and is effectively optimal. Similarly, for the maximum subset constraint sets, ADXOpt also performs very well, with an average gap of no more than 0.34%. However, with more complex constraints – namely the type 1 and type 2 precedence constraint sets – ADXOpt is no longer able to ensure high quality solutions. For example, for the type 1 precedence constraint set, with just $C = 2$ constraints with $B = 3$ products, the average optimality gap is 2.46%. In other cases, it can be much higher (e.g., for type 1 precedence constraints with $C = 2$ constraints and $B = 10$ products, the gap is 15.20%). With regard to timing, note that both approaches are fast and their speed is relatively insensitive to the presence of constraints. These results suggest that, while ADXOpt delivers high quality solutions under simple constraints, it leaves value on the table in the presence of complicated constraints. On the other hand, the MIO approach allows the firm to capture all of this lost value, at negligible computational cost.

With regard to our comparison here, we wish to emphasize that ADXOpt was not originally designed to accommodate complex constraints. Given this, one may claim that the manner in which we have adapted ADXOpt to handle constraints is not the most appropriate; that one may further modify the procedure to attain better results.

n	K	Constraint type	ADXOpt Gap (%)	MIO Time (s)	ADXOpt Time (s)
30	100	No constraints	0.07	0.22	0.07
30	100	Max. subset, $C = 2$, $B = 5$, $U = 3$	0.19	0.41	0.06
30	100	Max. subset, $C = 2$, $B = 10$, $U = 3$	0.34	0.59	0.03
30	100	Max. subset, $C = 3$, $B = 5$, $U = 3$	0.19	0.42	0.05
30	100	Max. subset, $C = 3$, $B = 10$, $U = 3$	0.34	0.58	0.03
30	100	Prec. type 1, $C = 1$, $B = 3$	0.72	0.21	0.06
30	100	Prec. type 1, $C = 2$, $B = 3$	2.46	0.37	0.05
30	100	Prec. type 1, $C = 1$, $B = 5$	1.92	0.37	0.06
30	100	Prec. type 1, $C = 2$, $B = 5$	6.31	0.54	0.04
30	100	Prec. type 1, $C = 1$, $B = 10$	7.44	0.58	0.03
30	100	Prec. type 1, $C = 2$, $B = 10$	15.20	1.31	0.02
30	100	Prec. type 2, $C = 3$, $B = 2$	0.47	0.23	0.07
30	100	Prec. type 2, $C = 5$, $B = 2$	0.68	0.26	0.06
30	100	Prec. type 2, $C = 10$, $B = 2$	1.17	0.25	0.04
30	100	Prec. type 2, $C = 3$, $B = 5$	1.64	0.25	0.05
30	100	Prec. type 2, $C = 5$, $B = 5$	1.98	0.21	0.03
30	100	Prec. type 2, $C = 10$, $B = 5$	6.35	0.12	0.01

Table 4.2: Results of constrained optimization comparison.

This is a fair criticism. The overall claim that we are making in this section is that our MIO modeling approach provides a simple and systematic way to accommodate a variety of constraints, allowing the user to obtain *provably* optimal solutions in operationally feasible times while *freeing the user* from the design of problem-specific algorithms. From this perspective, we do not believe that the aforementioned criticism invalidates our overall claim.

4.5.3 Estimation using column generation

To evaluate our estimation procedure, we proceed as follows. For different values of the number of products n, we consider different types of choice models. In particular, we consider the MMNL model with T classes, for which the choice probability $\mathbb{P}(i \mid S)$ is given by

$$\mathbb{P}(i \mid S) = \sum_{t=1}^{T} p_t \cdot \frac{\exp(u_{t,i})}{\sum_{i' \in S} \exp(u_{t,i'}) + \exp(u_{t,0})}. \tag{4.5}$$

We randomly generate our MMNL models as follows. For each option i and customer type t we generate the values $q_{t,i}$ uniformly on $[0, 1]$. For each customer type t, we then select four of the $n + 1$ utilities $u_{t,1}, \ldots, u_{t,n+1}$ randomly and set each chosen utility to $\log(Lq_{t,i})$, where L is a predefined value, and the remaining utilities we set to $\log(0.1q_{t,i})$. The mixing probabilities $p_1, \ldots, p_T$ are uniformly drawn from the $(T - 1)$-dimensional unit simplex. We indicate these models by MMNL(L, T); we consider $L \in \{5.0, 10.0, 100.0\}$ and $T \in \{1, 5, 10\}$.

For each number of products n and each type of choice model – value of T and L – we generate 100 random instances. For each instance, we randomly generate two sets of 100 assortments and compute the choice probabilities under the ground truth model for each option in each assortment. We treat the first set of 100 assortments as training data that we may use to fit the model according to the estimation procedure in Section 4.4, and the second set of 100 assortments as testing data that we will use to evaluate the predictive accuracy of our estimated models. We vary the number of assortments M that we use for training from the total 100 assortments and consider $M \in \{10, 20, 50, 100\}$. We run the estimation procedure and terminate it when the

training mean absolute error (MAE) is below 10^{-3}. To generate columns in the estimation procedure, we employ the local search procedure described in Section 4.4 with a maximum of ten repetitions; in all of our experiments, this was sufficient to find an entering column at each iteration. Using the resulting ranking-based model, we predict the choice probability of each option in each testing assortment, and compare each such probability to the corresponding true probability as per the ground truth model.

Table 4.3 reports the average training set and test set MAEs, where the average is taken over the 100 instances for each value of n and each generating model, with a training set of $M = 20$ assortments. We also report the average time required to run the procedure in seconds ("Est. Time (s)"), the average number of iterations ("Num. Iter."), the average number of rankings in the final model with non-zero probability (K) and the average value of P (recall that P is the dimension of the $\mathbf{v}$ vector). From this table, we can see that although the test set MAE is higher than the training MAE by approximately an order of magnitude, it is still quite good (on the order of 0.01 – 0.02). Moreover, the time to run the procedure is quite modest (in the largest cases, those with $n = 30$, no more than one minute on average).

We now turn our attention to the question of how the estimation procedure is affected by the amount of data. Table 4.4 shows how the accuracy and estimation time vary as the number of training assortments M varies, with n fixed to 30 and the utility scale parameter L fixed to 5.0. (As in Table 4.3, the results are averaged over the 100 instances for each case.) As the amount of data increases, the complexity of the underlying models increases (as measured by the average of the number of permutations K in the final model) and the test set MAE also decreases with additional data; in other words, the estimation procedure is able to learn more accurate models as the amount of data available increases. With regard to the time required to estimate the model, we find that even with 100 training assortments, the time required to run the procedure is still modest (no more than approximately five minutes on average).

Another interesting question to consider is how susceptible the procedure is to

n	Generating model	M	Train MAE	Test MAE	Est. Time (s)	Num. Iter.	K	P
10	MMNL(5.0, 1)	20	0.0008	0.0174	0.68	105.0	65.0	120.3
10	MMNL(5.0, 5)	20	0.0010	0.0200	0.57	90.6	69.6	119.2
10	MMNL(5.0, 10)	20	0.0010	0.0197	0.53	86.4	72.5	120.7
10	MMNL(10.0, 1)	20	0.0009	0.0162	0.71	102.1	62.4	119.3
10	MMNL(10.0, 5)	20	0.0010	0.0185	0.59	94.8	69.2	120.8
10	MMNL(10.0, 10)	20	0.0010	0.0195	0.50	87.1	72.3	120.2
10	MMNL(100.0, 1)	20	0.0007	0.0165	0.79	107.4	59.2	119.3
10	MMNL(100.0, 5)	20	0.0009	0.0189	0.67	104.2	67.8	120.1
10	MMNL(100.0, 10)	20	0.0010	0.0209	0.56	96.6	72.9	120.7
20	MMNL(5.0, 1)	20	0.0009	0.0127	7.41	220.4	125.2	219.9
20	MMNL(5.0, 5)	20	0.0010	0.0154	4.09	168.4	124.9	221.1
20	MMNL(5.0, 10)	20	0.0010	0.0159	3.30	144.9	121.4	220.0
20	MMNL(10.0, 1)	20	0.0009	0.0129	9.42	264.5	127.2	219.6
20	MMNL(10.0, 5)	20	0.0010	0.0153	4.96	194.7	128.9	220.2
20	MMNL(10.0, 10)	20	0.0010	0.0169	3.71	162.1	127.2	220.9
20	MMNL(100.0, 1)	20	0.0008	0.0120	11.13	281.1	117.3	221.0
20	MMNL(100.0, 5)	20	0.0010	0.0151	7.53	244.4	126.6	220.9
20	MMNL(100.0, 10)	20	0.0010	0.0174	5.00	196.6	131.2	219.4
30	MMNL(5.0, 1)	20	0.0010	0.0104	23.49	311.8	178.3	320.5
30	MMNL(5.0, 5)	20	0.0010	0.0124	13.83	219.3	168.2	317.9
30	MMNL(5.0, 10)	20	0.0010	0.0129	10.76	181.8	156.1	319.9
30	MMNL(10.0, 1)	20	0.0010	0.0100	32.52	382.9	188.5	321.8
30	MMNL(10.0, 5)	20	0.0010	0.0126	16.49	254.1	171.6	317.6
30	MMNL(10.0, 10)	20	0.0010	0.0137	13.20	215.1	167.0	318.0
30	MMNL(100.0, 1)	20	0.0009	0.0093	47.79	473.9	174.2	320.2
30	MMNL(100.0, 5)	20	0.0010	0.0126	44.20	371.9	177.6	324.9
30	MMNL(100.0, 10)	20	0.0010	0.0142	25.20	283.3	175.6	319.1

Table 4.3: Results of estimation procedure.

n	Generating model	M	Train MAE	Test MAE	Est. Time (s)	Num. Iter.	K	P
30	MMNL(5.0, 1)	10	0.0009	0.0174	4.65	162.3	99.3	161.0
30	MMNL(5.0, 1)	20	0.0010	0.0104	23.49	311.8	178.3	320.5
30	MMNL(5.0, 1)	50	0.0010	0.0054	103.12	489.8	288.4	802.2
30	MMNL(5.0, 1)	100	0.0010	0.0036	302.39	637.0	388.0	1605.8
30	MMNL(5.0, 5)	10	0.0010	0.0192	3.42	123.5	96.8	158.7
30	MMNL(5.0, 5)	20	0.0010	0.0124	13.83	219.3	168.2	317.9
30	MMNL(5.0, 5)	50	0.0010	0.0065	72.23	373.5	289.0	795.9
30	MMNL(5.0, 5)	100	0.0010	0.0043	231.02	524.9	414.1	1597.8
30	MMNL(5.0, 10)	10	0.0010	0.0190	3.09	107.8	93.7	160.3
30	MMNL(5.0, 10)	20	0.0010	0.0129	10.76	181.8	156.1	319.9
30	MMNL(5.0, 10)	50	0.0010	0.0073	68.88	345.1	291.3	799.6
30	MMNL(5.0, 10)	100	0.0010	0.0047	225.90	493.9	422.4	1601.0

Table 4.4: Results of estimation procedure as available data varies.

overfitting. To study this, we fix the number of training assortments M to 20 and we consider those instances with $n = 30$, $T = 10$ and $L = 5.0$. We run the estimation procedure and vary the training MAE tolerance; for each threshold we choose, we save the model at termination (the set of permutations and the probability distribution), and measure the MAE of this model on the test set. Table 4.5 displays the results from this experiment, averaged over the 100 instances corresponding to each case. From this table, we can see that although the test set MAE is generally higher than the training set MAE, it does not increase when the training MAE is further decreased; rather, it appears to converge. Figure 4-1 plots how the training MAE and test MAE evolve for one instance with the number of iterations to further emphasize the above point.

4.5.4 Comparison of revenue predictions

We now compare the predictive power of our approach with that of two other approaches. In the first approach, we fit an MNL model to a training set of assortments using maximum likelihood estimation and using this fitted model, we predict the expected revenue of each assortment in the test set. In the second approach, we con-

Train MAE Tol.	Train MAE	Test MAE	Time (s)	Num. Iter.	K
0.1000	0.0941	0.0961	0.1	3.5	3.5
0.0500	0.0464	0.0532	0.3	10.7	10.7
0.0100	0.0098	0.0214	1.9	38.1	37.4
0.0050	0.0049	0.0172	3.7	67.3	64.0
0.0010	0.0010	0.0131	11.0	183.6	158.3
0.0005	0.0005	0.0124	15.4	241.8	198.7
0.0001	9.83×10^{-5}	0.0117	24.7	340.1	264.4
5.00×10^{-5}	4.83×10^{-5}	0.0116	27.4	363.3	278.6
1.00×10^{-5}	8.58×10^{-6}	0.0115	30.3	388.1	293.6
5.00×10^{-6}	3.98×10^{-6}	0.0115	30.7	391.9	296.6
1.00×10^{-6}	4.24×10^{-7}	0.0115	31.2	395.6	299.6

Table 4.5: Results of estimation procedure as training MAE tolerance decreases. Results correspond to MMNL(5.0, 10) instances with $n = 30$ products and $M = 20$ training assortments.

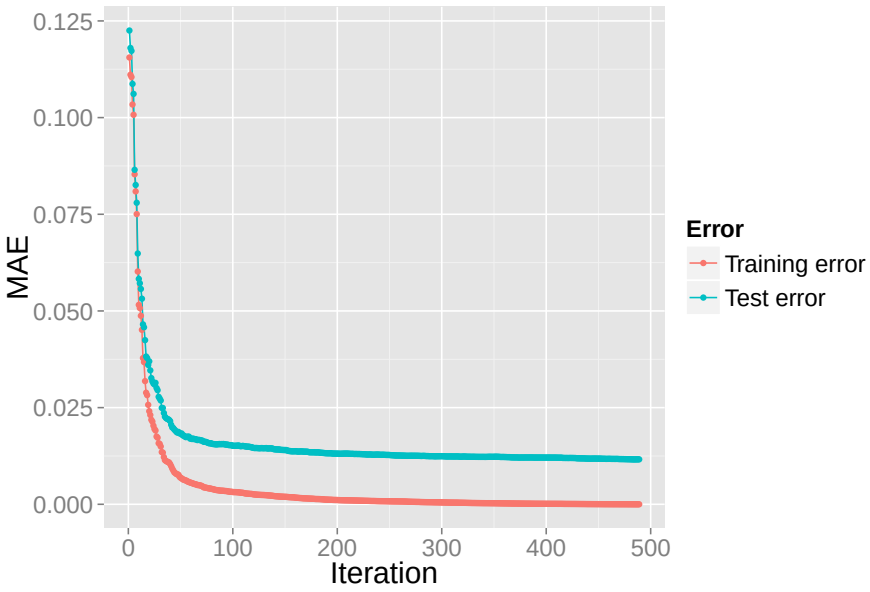

Figure 4-1: Evolution of training error and testing error with each column generation iteration for one MMNL instance with $n = 30$, $T = 10$, $L = 5.0$ and $M = 20$ training assortments.

sider the worst-case approach of [44] with the same training set of assortments and use it to make predictions of the expected revenue on the test set of assortments. Our implementation of the worst-case approach uses randomized sampling as described in [44]. This approach involves randomly sampling a subset of size K_{sample} of all of the possible permutations of the $n+1$ options and solving the sampled version of the worst-case problem. One of the difficulties of using sampling is that the sampled worst-case LO problem may turn out to be infeasible. To address this issue, we proceed as follows. For a given value of K_{sample}, we randomly sample K_{sample} rankings, and check if the sampled problem is feasible. If it is feasible, we use that sample to make all worst-case predictions on the test set. If it is not feasible, we sample again at the current value of K_{sample} and check again. If we encounter infeasibility after ten such checks, we move on to the next value of K_{sample}. The sequence of K_{sample} values that our procedure scans through is

$$1 \times 10^4, 2 \times 10^4, \ldots, 9 \times 10^4,\ 1 \times 10^5, 2 \times 10^5, \ldots, 9 \times 10^5,\ 1 \times 10^6.$$

This approach lead to a feasible set of sampled permutations $\mathcal{K}_{\text{sample}}$ in all of our numerical comparisons. Our goal in applying this procedure was to ensure that we find a feasible set of permutations quickly.

In addition to these three approaches, we also consider another version of our column generation approach, which we refer to as the "averaged" column generation approach. Recall that our implementation of our column generation procedure involves solving the subproblem using a local search, which starts from a random permutation. Given this, and given the fact that in general there may be many probability distributions $\boldsymbol{\lambda}$ that satisfy the linear system $\mathbf{A}\boldsymbol{\lambda} = \mathbf{v}$, repeated executions of the column generation procedure with the same data may produce different collections of rankings and probability distributions over them. Inspired by ensemble methods in statistical learning that combine multiple prediction methods into a single, more accurate prediction method (for further details see, e.g., [57]), we consider averaging the probability distributions produced by these repeated runs. Specifically,

			CG		AvgCG		MNL		WC		
n	Generating model	M	AE	ME	AE	ME	AE	ME	AE	ME	AT (s)
20	MMNL(5.0, 1)	20	1.72	13.07	1.16	8.28	0.00	0.00	3.83	16.93	0.56
20	MMNL(5.0, 5)	20	1.90	8.88	1.36	6.41	2.50	9.57	6.27	15.99	0.32
20	MMNL(5.0, 10)	20	1.77	7.40	1.29	5.22	2.05	7.91	11.83	23.19	0.44
20	MMNL(10.0, 1)	20	1.81	15.40	1.34	10.88	0.00	0.00	3.16	19.38	0.82
20	MMNL(10.0, 5)	20	1.98	9.69	1.51	7.35	2.79	11.64	4.78	15.16	0.36
20	MMNL(10.0, 10)	20	1.96	8.92	1.52	6.62	2.41	9.33	8.04	17.86	0.30
20	MMNL(100.0, 1)	20	1.82	16.41	1.36	12.27	0.00	0.00	2.61	21.19	2.64
20	MMNL(100.0, 5)	20	1.94	10.38	1.60	8.07	3.54	14.73	4.27	17.06	0.97
20	MMNL(100.0, 10)	20	2.07	9.19	1.71	7.25	2.88	11.49	5.75	15.61	0.39

Table 4.6: Results of revenue prediction comparison for $n = 20$ instances with $M = 20$ training assortments.

given τ runs of the column generation procedure and the corresponding probability distributions $\boldsymbol{\lambda}^1, \ldots, \boldsymbol{\lambda}^\tau$, we produce the averaged probability distribution $\boldsymbol{\lambda}^{\text{avg}}$ as

$$\boldsymbol{\lambda}^{\text{avg}} = \frac{1}{\tau} \sum_{i=1}^{\tau} \boldsymbol{\lambda}^i.$$

In our implementation of this averaged approach, we use the same column generation implementation as in Section 4.5.3 for $\tau = 10$ repetitions.

For each instance that we consider, we record the average absolute revenue error (AE) and maximum absolute revenue error (ME) over the assortments in the test set of that instance; we also record the average time (AT) in seconds to make a prediction, taken over the assortments in the test set. Table 4.6 reports the metrics for each prediction approach, averaged over the instances from Section 4.5.3 for $n = 20$ products and $M = 20$ training set assortments. (Note that we only report the AT metric for the worst case approach, as it was on the order of milliseconds for the other three approaches.) In the table and the discussion that follows, we use "CG", "AvgCG", "MNL" and "WC" to indicate the single pass column generation, averaged column generation, fitted MNL and worst-case methods, respectively.

With regard to MNL, we can see that when $T = 1$ – the ground truth model is a single-class MNL model – fitting an MNL model leads to essentially perfect predictions, as we would expect. However, for $T > 1$, the predictions become considerably

			CG		MNL	
n	Generating model	M	AE	ME	AE	ME
20	MMNL(100.0, 5)	10	3.40	16.38	3.67	15.30
20	MMNL(100.0, 5)	20	1.94	10.38	3.54	14.73
20	MMNL(100.0, 5)	50	0.78	6.00	3.40	14.45
20	MMNL(100.0, 5)	100	0.41	4.50	3.37	14.24
20	MMNL(100.0, 10)	10	3.35	13.61	3.03	12.15
20	MMNL(100.0, 10)	20	2.07	9.19	2.88	11.49
20	MMNL(100.0, 10)	50	0.90	5.03	2.81	11.11
20	MMNL(100.0, 10)	100	0.47	3.62	2.78	10.96

Table 4.7: Results of revenue prediction comparison between CG and MNL approaches for $n = 20$ instances with $L = 100.0$, $T \in \{5, 10\}$, as number of training assortments M varies.

less accurate and in particular, are less accurate than those produced by CG. It is also interesting to compare MNL to CG as the amount of data (the number of training assortments M) varies. Table 4.7 presents the same accuracy metrics as Table 4.6 as M varies in $\{10, 20, 50, 100\}$ for $L = 100.0$ and $T \in \{5, 10\}$. From this table, we can see that while the prediction error of CG decreases significantly as M increases, the error of the fitted MNL model decreases only slightly. In words, the MNL approach is not able to learn the true model with additional data, because it is constrained to a single parametric form, while our approach is able to learn from this additional data.

With regard to the worst-case approach, we see from Table 4.6 that over all of the types of instances – values of L and T – the column generation approach yields significantly more accurate predictions of revenue than the worst-case approach of [44]. For example, for $L = 5.0$ and $T = 10$, the average revenue error, averaged over 100 instances, is 1.77 for the column generation approach, while for the worst-case approach it is 11.83 – approximately an order of magnitude higher. Moreover, since each revenue prediction in the worst-case approach involves solving a large-scale LO problem, the average time to make predictions is considerably higher; the largest average time is on the order of 2.6 seconds per prediction, whereas for both CG and the force-fitted MNL model it is on the order of milliseconds.

Why does the approach of [44] give less accurate predictions? There are two rea-

sons for this. The first reason is that the worst-case approach predicts the lowest possible revenue; it finds the probability distribution that is consistent with the data and that results in the lowest possible revenue. Depending on the nature of the choice model and the data, the set of probability distributions that are consistent with the data may be quite large and the revenue predictions may thus be quite conservative. The second reason for the inaccuracy is that the worst-case approach predicts the lowest possible revenue *for each assortment*. Although each such prediction arises from a probability distribution that is consistent with the data, the revenue-minimizing probability distribution *may not be the same for each assortment*; thus, the ensemble of revenue predictions produced by the worst-case approach may not be realized by a *single* choice model. Due to this property, we therefore expect that the worst-case approach may exhibit some error in the aggregate that cannot be avoided.

Lastly, we can see that averaging leads to more accurate predictions. In particular, from Table 4.6, we can see that AvgCG in all cases leads to lower average and maximum errors than CG. This improvement can be understood from a statistical learning perspective by considering bias and variance; namely, averaging reduces the variance in the mean square error made by CG that is induced by the randomness of the estimation procedure, without changing the bias.

4.5.5 Combining estimation and optimization

We now consider combining our estimation approach with our optimization approach. To evaluate this combined approach for a given instance, we first estimate the rankings $\sigma^1, \ldots, \sigma^K$ and probabilities $\lambda^1, \ldots, \lambda^K$ from the data for that instance using our column generation procedure, and we then use these estimated rankings and probabilities to formulate and solve problem (4.1), yielding an assortment. We then evaluate the *true* revenue of the assortment under the model that generated the data of that instance and compare it to the optimal revenue for that generating model. Using $R_{\text{true}}(S)$ to denote the true revenue of the assortment S and R^*_{true} to denote

the optimal true revenue, we define the optimality gap G as

$$G = 100\% \times \frac{R^*_{\text{true}} - R_{\text{true}}(S)}{R^*_{\text{true}}}. \tag{4.6}$$

The smaller G is, the better our approach performs; a value of 0% indicates that our approach captures all of the revenue that is possible under the model that generated the data.

Table 4.8 presents results for our approach for the same instances considered in Section 4.5.3. (As in our other comparisons, we report the averages of each metric over the 100 instances for each value of n and generating model. We also re-use the same rankings and probabilities that were estimated in Section 4.5.3 for Table 4.3.) From this table, we can see that our approach results in an optimality gap on the order of a few percent, using only $M = 20$ training assortments. We can also see that the total time for the combined approach – estimating the rankings and probabilities from the transaction data and then solving the MIO – is also quite small; in the most extreme case, it is no more than 50 seconds on average.

How does the optimality gap change as the data increases? Table 4.9 shows the average optimality gap for the instance set from Section 4.5.3, restricted to $n = 30$ and $L = 5.0$, as M varies in $\{10, 20, 50, 100\}$. We can see that as the amount of data increases, the average optimality gap decreases. At $M = 10$ training assortments, it is on average 3-4%, but decreases to below 1.5% with $M = 100$ assortments. This table complements Table 4.4 in Section 4.5.3, which showed that increasing data led to more accurate out-of-sample predictions; here, we have shown that more data translates to making more effective decisions.

Finally, we consider the question of whether it is possible to overfit the model from the perspective of optimization: namely, if we fit the training data too much, will we make worse decisions? Table 4.10 shows how the average optimality gap varies as the training MAE tolerance is decreased for those instances corresponding for $n = 30$, $T = 10$ and $L = 5.0$ from Section 4.5.3. We can see that as the tolerance decreases, the optimality gap on average in general decreases. Although the relationship is

n	Generating model	M	Gap (%)	Est. Time (s)	Opt. Time (s)	Total Time (s)
10	MMNL(5.0,1)	20	2.98	0.68	0.02	0.70
10	MMNL(5.0,5)	20	1.98	0.57	0.02	0.59
10	MMNL(5.0,10)	20	1.62	0.53	0.02	0.55
10	MMNL(10.0,1)	20	1.87	0.71	0.02	0.73
10	MMNL(10.0,5)	20	1.43	0.59	0.02	0.61
10	MMNL(10.0,10)	20	2.10	0.50	0.02	0.53
10	MMNL(100.0,1)	20	2.86	0.79	0.02	0.81
10	MMNL(100.0,5)	20	1.38	0.67	0.02	0.69
10	MMNL(100.0,10)	20	2.09	0.56	0.02	0.58
20	MMNL(5.0,1)	20	4.24	7.41	0.16	7.57
20	MMNL(5.0,5)	20	2.39	4.09	0.13	4.22
20	MMNL(5.0,10)	20	1.98	3.30	0.12	3.42
20	MMNL(10.0,1)	20	2.92	9.42	0.12	9.54
20	MMNL(10.0,5)	20	1.93	4.96	0.12	5.08
20	MMNL(10.0,10)	20	2.47	3.71	0.16	3.87
20	MMNL(100.0,1)	20	3.57	11.13	0.09	11.22
20	MMNL(100.0,5)	20	3.01	7.53	0.14	7.67
20	MMNL(100.0,10)	20	1.80	5.00	0.13	5.13
30	MMNL(5.0,1)	20	2.94	23.49	0.39	23.88
30	MMNL(5.0,5)	20	2.10	13.83	0.39	14.22
30	MMNL(5.0,10)	20	2.30	10.76	0.47	11.22
30	MMNL(10.0,1)	20	3.51	32.52	0.43	32.95
30	MMNL(10.0,5)	20	2.45	16.49	0.50	16.98
30	MMNL(10.0,10)	20	2.00	13.20	0.49	13.69
30	MMNL(100.0,1)	20	4.35	47.79	0.29	48.09
30	MMNL(100.0,5)	20	2.20	44.20	0.71	44.90
30	MMNL(100.0,10)	20	2.43	25.20	0.65	25.86

Table 4.8: Results of combining the estimation and optimization procedures over a wide range of MMNL models.

n	Generating model	M	Gap (%)	Est. Time (s)	Opt. Time (s)	Total Time (s)
30	MMNL(5.0,1)	10	3.85	4.65	0.19	4.84
30	MMNL(5.0,1)	20	2.94	23.49	0.39	23.88
30	MMNL(5.0,1)	50	2.05	103.12	0.76	103.87
30	MMNL(5.0,1)	100	1.47	302.39	1.08	303.47
30	MMNL(5.0,5)	10	3.45	3.42	0.19	3.61
30	MMNL(5.0,5)	20	2.10	13.83	0.39	14.22
30	MMNL(5.0,5)	50	1.38	72.23	0.88	73.11
30	MMNL(5.0,5)	100	0.98	231.02	1.62	232.64
30	MMNL(5.0,10)	10	3.50	3.09	0.26	3.35
30	MMNL(5.0,10)	20	2.30	10.76	0.47	11.22
30	MMNL(5.0,10)	50	1.36	68.88	1.16	70.03
30	MMNL(5.0,10)	100	0.78	225.90	2.22	228.12

Table 4.9: Results of combining the estimation and optimization procedures as the amount of available data (the number of training assortments M) varies.

not monotonic, the optimality gap does not dramatically worsen when the column generation procedure is used to fit the training data to a very high level of precision. To provide a better visualization of this relationship, we show in Figure 4-2 how the optimality gap varies with the number of column generation iterations for a single instance (the same one studied in Figure 4-1).

4.5.6 Comparison of combined estimation and optimization procedure

Finally, we compare our combined estimation-optimization approach with other approaches. For each instance that we consider, we run the corresponding optimization procedure for each choice model that we considered in Section 4.5.4: for our estimated finite permutation model, we solve the MIO problem (4.1); for the fitted MNL model, we find the revenue-ordered subset with the highest predicted revenue [89]; and for the worst-case approach, we apply the ADXOpt heuristic of [62] with at most one removal for each product. As in Section 4.5.4, we consider optimizing the the ranking-based model produced by a single pass of the column generation procedure

Train MAE Tol.	Gap (%)	Est. Time (s)	Opt. Time (s)	Total Time (s)
0.1000	20.95	0.06	0.00	0.06
0.0500	10.87	0.29	0.01	0.30
0.0100	4.58	1.94	0.06	2.01
0.0050	2.94	3.70	0.14	3.84
0.0010	2.06	11.04	0.49	11.53
0.0005	2.11	15.42	0.59	16.00
0.0001	1.87	24.74	0.88	25.62
5.00×10^{-5}	1.86	27.36	0.93	28.29
1.00×10^{-5}	1.87	30.27	0.93	31.19
5.00×10^{-6}	1.90	30.72	1.01	31.74
1.00×10^{-6}	1.87	31.16	0.98	32.14

Table 4.10: Results of combining the estimation and optimization procedures as training MAE tolerance varies. Results correspond to MMNL(5.0,10) instances with $n = 30$ products and $M = 20$ training assortments.

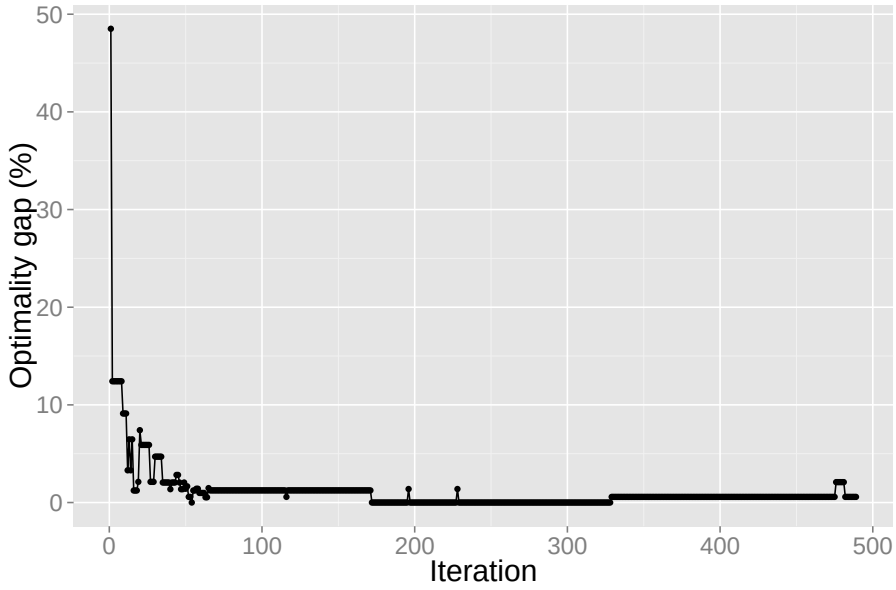

Figure 4-2: Evolution of optimality gap with each column generation iteration for one MMNL instance with $n = 30$, $T = 10$, $L = 5.0$ and $M = 20$ training assortments.

as well as optimizing the averaged ranking-based model that results from ten runs of the column generation procedure.

For the final assortment produced by each approach, we compute the gap of this assortment relative to the *true* optimal revenue for the underlying ground truth model. We record the total time required for each approach: for our approach, we record the total of the time required for estimation (using the procedure in Section 4.4) and optimization using the MIO formulation (4.1); for the MNL approach, we record the total of the time required for maximum likelihood estimation and optimization via enumeration of the revenue-ordered subsets; and for the worst-case approach, we record the time required for optimization using ADXOpt, which includes the time expended each time that the worst-case revenue is computed. Each objective function evaluation for the worst-case/ADXOpt approach involves solving the worst-case model from [44], which is a large-scale LO problem. We solve it by solving the sampled problem using the same sampled permutations that were used to make the revenue predictions in Section 4.5.4.

Table 4.11 reports the results of this comparison, which are averaged over the 100 instances corresponding to each value of n and each generating model. In the table and in the discussion that follows, we use "CG+MIO", "AvgCG+MIO", "MNL" and "WC+ADXOpt" to indicate the single pass column generation and MIO combination, the averaged column generation and MIO combination, the MNL approach and the worst-case and ADXOpt combination, respectively.

With regard to WC+ADXOpt, CG+MIO is generally better and in a number of cases significantly so. For example for $n = 20$, $L = 5.0$, $T = 10$, the MIO approach achieves revenues that are on average 2.0% from the true optimal value, while the worst-case/ADXOpt approach achieves revenues that are on average 13.7% below the true optimal value. In one case ($L = 100.0$, $T = 1$) the worst-case/ADXOpt approach yields a lower gap, although the difference is quite small (3.31% compared to 3.57% for our approach). Furthermore, with regard to the running time, our estimation and MIO procedure together require strikingly less time than the worst-case approach of [44] and the ADXOpt local search heuristic of [62]; in our approach, the average time

n	Generating model	M	CG+MIO Gap (%)	CG+MIO Time (s)	AvgCG+MIO Gap (%)	AvgCG+MIO Time (s)	MNL Gap (%)	MNL Time (s)	WC+ADXOpt Gap (%)	WC+ADXOpt Time (s)
20	MMNL(5.0,1)	20	4.24	7.57	2.18	73.49	0.00	0.69	4.46	314.79
20	MMNL(5.0,5)	20	2.39	4.22	0.96	48.58	3.88	0.73	5.26	177.34
20	MMNL(5.0,10)	20	1.98	3.42	0.92	37.10	3.25	0.71	13.65	250.50
20	MMNL(10.0,1)	20	2.92	9.54	0.97	107.00	0.00	0.70	6.30	463.60
20	MMNL(10.0,5)	20	1.93	5.08	1.20	54.21	4.27	0.72	4.17	193.51
20	MMNL(10.0,10)	20	2.47	3.87	1.01	43.71	3.55	0.74	6.17	171.88
20	MMNL(100.0,1)	20	3.57	11.22	0.96	118.24	0.00	0.76	3.31	1432.92
20	MMNL(100.0,5)	20	3.01	7.67	1.43	82.41	7.89	0.79	3.72	573.27
20	MMNL(100.0,10)	20	1.80	5.13	1.04	55.29	5.03	0.80	4.10	212.08

Table 4.11: Results of comparison of combined estimation-optimization approaches for $n = 20$, MMNL$(\cdot, \cdot)$ instances.

required to both estimate the ranking-based model and optimize it is no more than 12 seconds in the most extreme case, whereas for the worst-case/ADXOpt approach, the average time is usually on the order of 200 seconds, and in one collection of instances ($L = 10.0$, $T = 5$) more than 20 minutes on average.

With regard to the MNL approach, we see that for the cases where $T = 1$ – the ground truth is a single-class MNL model – fitting an MNL model to the data and optimizing it results in an optimality gap of 0%, as we would expect. However, for $T > 1$, we can see that fitting and optimizing MNL can be quite suboptimal, and performs worse than our approach; for example with $L = 100.0$, $T = 5$, the average optimality gap of our approach is 3.01% compared to 7.89% for the MNL approach. To further compare the MNL approach to ours, we can also consider how the optimality gap changes as the number of training assortments M is varied. Table 4.12 shows how the optimality gap of the MNL approach and our approach vary as M varies for a subset of instances. Analogously to Table 4.7, which showed that the MNL approach does not become more accurate with more data, this table shows that the MNL approach does not result in more optimal decisions with more data. In contrast, our approach leads to improved decisions with more data.

Lastly, we observe that optimizing the averaged model (AvgCG+MIO) leads to even better performance than optimizing a model from a single column generation run (CG+MIO). For example, with $L = 10.0$ and $T = 10$, averaging reduces the gap from

n	Generating model	M	CG+MIO Gap (%)	CG+MIO Time (s)	MNL Gap (%)	MNL Time (s)
20	MMNL(100.0,5)	10	5.39	1.34	8.21	0.74
20	MMNL(100.0,5)	20	3.01	7.67	7.89	0.79
20	MMNL(100.0,5)	50	1.54	42.39	8.07	0.91
20	MMNL(100.0,5)	100	0.98	116.41	7.47	1.10
20	MMNL(100.0,10)	10	3.58	1.04	5.47	0.75
20	MMNL(100.0,10)	20	1.80	5.13	5.03	0.80
20	MMNL(100.0,10)	50	0.88	34.94	4.76	0.91
20	MMNL(100.0,10)	100	0.53	95.72	5.05	1.11

Table 4.12: Results of optimality gap comparison between MNL and CG+MIO approaches as number of training assortments M varies, for $n = 20$, MMNL instances with $L = 100.0$ and $T \in \{5, 10\}$.

2.47% to 1.01%. This improvement is particularly noteworthy given the relatively low amount of data that was available for building the model. Comparing AvgCG+MIO to the other two methods, we can see that AvgCG+MIO delivers significantly better performance than WC+ADXOpt in all instances and MNL in all instances with $T > 1$. Based on these results, we believe that our approach has the potential to capture significant value in practical assortment decisions in the presence of limited data.

4.6 Conclusions

In this chapter, we have presented a practical method for transforming limited historical transaction data into effective assortment decisions. Our method consists of two pieces: an estimation procedure for extracting a flexible, generic choice model from the data and an assortment optimization procedure for finding the best assortment given the estimated choice model. Modern mathematical optimization plays a key role in both pieces: the estimation piece is based on efficiently solving a large-scale LO problem using column generation, while the assortment optimization piece is based on solving a practically tractable MIO problem. We show that our methodology is scalable, flexible, leads to accurate revenue predictions and leads to near-optimal assortments that outperform alternative parametric and non-parametric approaches.

There are a number of promising directions for future research. In this work we have assumed that the estimation occurs only once before the assortment decision is made, which is also made only once. However, in a practical setting, one may make multiple assortment decisions over time: each assortment decision will yield new data on the behavior of the market, allowing the firm to change the assortment over time in response to this data. Thus, framing the problem in a dynamic setting with learning is a valuable next step. In a different direction, one may consider how to extend the procedure when the data is richer and more fine-grained, and when the decision of the firm is at a similar resolution: for example, a firm may track transactions made by individual customers and attributes of those customers, and may be able to make assortment decisions that are targeted to individual customers. The challenge in this setting is to estimate a model that predicts the choice probability of each item in an assortment *given a particular customer's attributes*, and to then use this model to optimally target customers. We consider this problem from a different angle in the next chapter.

Chapter 5

Personalized assortment planning via recursive partitioning

5.1 Introduction

Personalization refers to making operational decisions that are tailored to specific individuals. Personalization is used in a variety of business settings, most notably in online retail. Personalization is enabled by technological advancements that both allow for the collection of data at the individual level as well as the ability to make decisions at the resolution of individuals.

The problem of personalized assortment planning is to decide which products to offer to the customer, based on information about that customer. The critical prerequisite for making such decisions is data. A firm may be in possession of large volumes of data on historical transactions. Such data, in its most basic form, will indicate who the customer is, what products the customer was offered and what the customer ultimately purchased. By using this data to learn about the preferences of individual customers, the firm can make pricing and assortment decisions that are tailored to individuals.

The central challenge in personalization lies in effectively leveraging the data. In particular, for each customer, the ideal situation would be to possess abundant data on prior transactions of that customer, to use this data to infer the preferences of

the customer and to then make an effective assortment decision. In practice, this is not the case. For a given customer, we may have some prior purchasing data, but typically this data on its own is insufficient to make a conclusive statement about the customer's preferences or to make a sound pricing/assortment decision. However, we have such "grains" of data for many customers. Some of these customers may be very similar to the given customer with respect to their attributes (e.g., the same ZIP code, age, browser, prior browsing behavior, and so on); as a result, the purchasing data of those customers could serve in lieu of "true" data generated by the customer of interest. Other customers may be quite different; although those customers also provide purchasing data, this data should be discounted with regard to constructing the preferences of the customer of interest.

In this chapter, we present an approach for making personalized assortment decisions from data. The approach involves dividing the customer population into a group of mutually disjoint segments, with the choice behavior of each segment being captured by a choice model estimated from the transactions that correspond to that segment. The segments are constructed by recursively partitioning the customer population according to the attributes.

We make the following two contributions:

1. We present an approach based on recursive partitioning for building a customer-level choice model and thus for making personalized assortment decisions. This approach has two major benefits. First, by partitioning the customer attribute space in this way, the approach has the potential to automatically capture complex, non-linear relationships between the choice behavior of the customers and the customer attributes. Second, the model can be represented through a tree, where each non-leaf node in the tree represents a split along a customer attribute; in this way, the segments that comprise the model can be easily interpreted and can provide managerial intuition for how different groups of customers choose.

2. Using synthetic data, we numerically compare our tree-based personalized as-

sortment method to the classical "uniform" assortment strategy, where one ignores the customer attribute data in estimating the choice model and offers the same assortment to all customers. We show that our tree-based method gives stronger decisions than the uniform strategy and achieves revenues that are moderately close to the full information optimal (where one has access to the true customer-level choice model).

The rest of this chapter is organized as follows. In Section 5.2, we provide a brief overview of the relevant literature. In Section 5.3, we present the high-level model of customers and choice behavior that we will assume for our method. In Section 5.4 we define our recursive partitioning method. In Section 5.5, we present the results of a small computational experiment to show the benefit of using our partitioning method. Finally, in Section 5.6, we conclude.

5.2 Literature review

Personalization is a relatively new topic in the assortment optimization literature and more broadly in operations management. There are two major streams of papers in this area within operations management. One is focused on dynamic assortment planning. In this setting, a retailer faces an unknown, stochastic sequence of customers and has to decide what assortment of products to offer to each customer that arrives so as to maximize revenue; the customers are of different types, and the retailer has a limited inventory of each product. [11] considers this problem in the case where each customer type corresponds to a different multinomial logit choice model; they derive structural properties of the optimal policy in a specific case and propose a heuristic based on their insights. [63] proposes a re-optimization scheme for choice-based network revenue management that accounts for different customer types. [53] proposes an algorithm called inventory balancing that performs well both when the arrival of customer types follows a known, stochastic process, but also when the customer sequence may be chosen in an adversarial way. In this body of work, one presumes access to a predictive model that maps a customer to a choice model, and the chal-

lenge lies in the dynamic nature of the problem: do we offer a scarce product to the customer at hand to potentially obtain revenue, or do we forgo the revenue and save it for a later customer who may be more "picky"? In contrast, in our work, the focus is on actually building the mapping from customers to choice models and ultimately to an assortment, rather than on making good decisions dynamically with inventory constraints.

The second stream, which is closer to our work, is based more on estimation. One notable recent paper is [34], which proposes modeling customers by using a customer-level logit model, where the product utilities are assumed to be linear in the attributes of the customer; to obtain a decision for a customer, one computes the product utilities for the current customer, and then finds the assortment that maximizes revenue for that customer's specific logit model. The paper develops guarantees on out-of-sample prediction quality and revenues, and also shows that the algorithm provides good performance in simulated data. This chapter considers the same problem, but instead of assuming the utilities are linearly related to the customer attributes, we model the utilities in a partially non-parametric way: customers still choose according to an MNL model, but the product utility vector defining that MNL model is a piecewise constant function of the customer attributes. This yields two benefits. The first is that it can potentially capture more complicated, non-linear relationships between the product utilities and customer attributes in an automatic way (i.e., without the modeler having to iteratively test different nonlinear transformations of the original customer attributes). The second benefit is that our method produces a partitioning of the customers that is amenable to interpretation; a firm can examine the segments that arise from our method and potentially obtain insight into how customer characteristics translate to choice behavior, which may be more challenging with a linear model.

Outside of operations management, the recursive partitioning method we propose is related to recursive partitioning as it is used in statistics and machine learning (see [29]). Within the statistics and machine learning literature, our method is very closely related to the idea of model-based (MOB) trees [97]. Like in regular classification

trees, in a MOB tree one attempts to predict a dependent variable Y using two sets of decision variables, $\mathbf{x}$ and $\mathbf{z}$ – however, rather than performing traditional splits using $\mathbf{x}$ and $\mathbf{z}$, one splits only on the variables $\mathbf{z}$. Then, in each leaf of the resulting tree, one builds a parametric model to predict Y using the variables $\mathbf{x}$. We shall see that this is similar to the strategy that we will use in building our predictive model: the customer attributes for each transaction are the $\mathbf{z}$ variables, which we split on, and we then attempt to fit an MNL model in each leaf to predict the choice (Y) from the assortment ($\mathbf{x}$). The difference between our approach and that of [97] is that the criterion we use to select the split is not parameter instability, but more simply the log likelihood. It would be interesting to compare and contrast different splitting criterions in future research. To the best of our knowledge, MOB trees where each leaf corresponds to an MNL model have not been used previously in the marketing and operations management literatures.

Lastly, the method we propose is related to the predictive-prescriptive framework of [16]. In this paper, the authors propose a method for optimizing the expected cost $\mathbb{E}[c(z;Y)]$, where c depends on the decision z and an unknown dependent variable Y, and where one has access to auxiliary/contextual information X that can be used to predict Y. They show that decisions from their method asymptotically converge to the decision corresponding to the full information optimum (where one has exact access to the exact conditional expectation $\mathbb{E}[Y \,|\, X = x]$). The setting in [16] is closely related to the setting we study here: the customer attributes in our setting are analogous to the contextual information X. The key difference between the problem we study here and the one in [16] is that, in [16], one assumes that one can directly observe the Y variable and that it is unaffected by the decision z. In our setting, this is not the case, because the choice we observe in each transaction is affected by the decision (the assortment) that was made in that transaction. One avenue to directly applying [16] to our problem would be if we could observe not the choice given the assortment, but rather the *ranking* the customer used to make his choice from the assortment; this ranking would then be the dependent variable Y we would be trying to predict. Unfortunately, this type of information is not accessible in most settings.

Bridging the framework of [16] to decisions involving customer choice is an interesting and important direction of future research.

5.3 Model

We begin by defining the underlying customer choice model and providing additional definitions in Section 5.3.1. In Section 5.3.2, we define the uniform assortment decision paradigm, where the firm offers the same assortment to all customers. Then, in Section 5.3.3, we define the paradigm of personalized assortment decisions, where the firm offers a different assortment to each customer based on their attributes.

5.3.1 Background

We assume that there are n products, indexed from 1 to n, that may be offered to the customer population. We assume, as in Chapter 4, that the index 0 is used to represent the no-purchase option. When offered an assortment $S \subseteq \{1, \ldots, n\}$, a customer may choose any product i from S, or may choose the no-purchase option 0. We use r_i to denote the marginal revenue of product $i \in \{1, \ldots, n\}$.

We assume that each customer is represented by a vector $\mathbf{c} = (c_1, \ldots, c_m)$ of binary attributes, that is, each $c_j \in \{0, 1\}$. We let $\mathcal{C} \subseteq \{0, 1\}^m$ denote the set of possible customer types (binary vectors). For each customer attribute vector $\mathbf{c}$, we assume that there is a probability $\mu(\mathbf{c})$ that a random customer from the population has attribute vector $\mathbf{c}$.

To define the choice behavior of the population, we let $\mathbb{P}(i \,|\, S; \mathbf{c})$ denote the probability that a random customer chooses the option $i \in S \cup \{0\}$ when offered the assortment S, conditional that they exhibit the attributes $\mathbf{c}$. We let $\mathbb{P}(i \,|\, S)$ denote the probability that a random customer chooses the option $i \in S \cup \{0\}$ when offered the assortment S, unconditioned on the attributes of the customer. The unconditional

choice probability $\mathbb{P}(i \mid S)$ can be written as

$$\mathbb{P}(i \mid S) = \sum_{\mathbf{c} \in \mathcal{C}} \mu(\mathbf{c}) \cdot \mathbb{P}(i \mid S; \mathbf{c}).$$

5.3.2 Uniform assortment decisions

In the uniform assortment setting, we ignore the fact that there exist customers who differ in their attributes and their choice behavior. We find the best assortment with respect to $\mathbb{P}(\cdot \mid \cdot)$:

$$S^*_{unif} = \arg \max_{S \subseteq \{1,\ldots,n\}} \sum_{i \in S} r_i \cdot \mathbb{P}(i \mid S),$$

and we offer the assortment S^*_{unif} to *all* customers (hence the name "uniform"; all customers are offered the same assortment). Under such a scheme, the expected per-customer revenue is given by

$$\begin{aligned} R^*_{unif} &= \sum_{i \in S^*_{unif}} r_i \cdot \mathbb{P}(i \mid S^*_{unif}) \\ &= \sum_{\mathbf{c} \in \mathcal{C}} \mu(\mathbf{c}) \cdot \left[\sum_{i \in S^*_{unif}} r_i \cdot \mathbb{P}(i \mid S^*_{unif}; \mathbf{c}) \right]. \end{aligned}$$

5.3.3 Personalized assortment decisions

In a personalized assortment setting, we would like to be able to change the assortment based on the attributes of the customer. To do this, we find the best assortment for each customer $\mathbf{c}$ in $\mathcal{C}$:

$$S^*(\mathbf{c}) = \arg \max_{S \subseteq \{1,\ldots,n\}} \sum_{i \in S} r_i \cdot \mathbb{P}(i \mid S\,;\, \mathbf{c}).$$

We now proceed to offer the assortment $S^*(\mathbf{c})$ to each customer $\mathbf{c}$. Under this personalized approach, the expected per-customer revenue is given by

$$R^*_{pers} = \sum_{\mathbf{c} \in \mathcal{C}} \mu(\mathbf{c}) \cdot \left[\sum_{i \in S^*(\mathbf{c})} r_i \cdot \mathbb{P}(i \mid S^*(\mathbf{c}); \mathbf{c}) \right].$$

The following simple result shows that the personalized decision $\{S^*(\mathbf{c})\}_{\mathbf{c}\in\mathcal{C}}$ always yields revenues that are at least as high as those from the uniform decision S^* that is offered to all customers:

Proposition 10 $R^*_{pers} \geq R^*_{unif}$.

Proof: Observe that for each $\mathbf{c} \in \mathcal{C}$, we have that

$$\sum_{i\in S^*(\mathbf{c})} r_i \cdot \mathbb{P}(i \mid S^*(\mathbf{c}); \mathbf{c}) \geq \sum_{i\in S^*} r_i \cdot \mathbb{P}(i \mid S^*_{unif}; \mathbf{c}),$$

because $S^*(\mathbf{c})$ maximizes $\sum_{i\in S} r_i \cdot \mathbb{P}(i \mid S; \mathbf{c})$ as a function of S. In words, the optimal decision for customer $\mathbf{c}$ yields a revenue from that customer that is at least as good as the revenue that the uniform decision would extract from that customer. We now have that

$$\begin{aligned} R^*_{pers} &= \sum_{\mathbf{c}\in\mathcal{C}} \mu(\mathbf{c}) \cdot \left[\sum_{i\in S^*(\mathbf{c})} r_i \cdot \mathbb{P}(i \mid S^*(\mathbf{c}); \mathbf{c}) \right] \\ &\geq \sum_{\mathbf{c}\in\mathcal{C}} \mu(\mathbf{c}) \cdot \left[\sum_{i\in S^*} r_i \cdot \mathbb{P}(i \mid S^*; \mathbf{c}) \right] \\ &= R^*_{unif}, \end{aligned}$$

as required. □

5.4 The proposed method

In Section 5.3, we presented the traditional uniform and the personalized assortment planning approaches, and we showed that the personalized assortment planning approach leads to provably higher revenues than the traditional approach. The prerequisite for applying the personalized assortment planning approach is the customer-level choice model, that is, a specification of $\mathbb{P}(i \mid S; \mathbf{c})$ for each assortment $S \subseteq \{1, \ldots, n\}$, option $i \in S \cup \{0\}$ and customer attribute vector $\mathbf{c} \in \mathcal{C}$. In reality, we do not know the true customer-level model; often, we do not even know the aggregated choice

probabilities $\mathbb{P}(i \mid S)$. In this section, we present a methodology for building an approximation of such a model from data. In Section 5.4.1, we begin by describing the type of data that we will use to build the model. We then present our recursive partitioning approach in Section 5.4.2.

5.4.1 Data

We assume that our historical transaction data consists of T previous transactions. Each transaction $t \in \{1, \ldots, T\}$ corresponds to a single customer with attribute vector $\mathbf{c}^t \in \mathcal{C}$ being offered the assortment $\mathcal{S}^t \subseteq \{1, \ldots, n\}$ and choosing option $p^t \in \mathcal{S}^t \cup \{0\}$.

As an example, consider an online retail setting where we have $n = 12$ products that we can potentially sell. With regard to the customers, we track four different attributes: IsWestCoast (whether the customer is in Washington/Oregon/California), IsMale (self-explanatory), IsChromeUser (whether the customer uses the Google Chrome web browser) and IsSafariUser (whether the customer uses the Apple Safari web browser). The customer attribute vector $\mathbf{c}$ consists of these attributes:

$$\mathbf{c} = (\text{IsWestCoast}, \text{IsMale}, \text{IsChromeUser}, \text{IsSafariUser}).$$

In this example, we have observed 18 transactions, which are shown in Table 5.1. As an example of how to read this table, consider transaction $t = 12$. In this transaction, the customer is a West Coast female user who uses Chrome (and not Safari); she was offered products 4, 6, 11 and 12, and she purchased product 4. As another example, in transaction 9, the user – a non-West Coast female Chrome user – was offered products 1, 5 and 11, and opted to not purchase anything ($p^t = 0$ for this transaction).

Note that in general, the data may be far more complicated than this. For example, the number of products and the number of unique assortments that appear in the data may be quite large. Similarly, the dimension of the customer attribute vector may be much larger than in this stylized example.

Transaction ID (t)	Customer attributes ($\mathbf{c}^t$)	Offer set ($\mathcal{S}^t$)	Choice (p^t)
1	(0,0,1,0)	$\{1,3,5,6\}$	0
2	(0,0,1,0)	$\{2,3,8\}$	8
3	(1,0,1,0)	$\{1,2,3,4,5,6,7,8,9,10,11,12\}$	9
4	(1,1,0,1)	$\{4,6,11,12\}$	4
5	(1,1,0,1)	$\{2,3,4,11,12\}$	4
6	(1,1,0,1)	$\{1,5,11\}$	11
7	(0,1,0,1)	$\{1,5,11\}$	0
8	(0,0,1,0)	$\{1,5,11\}$	5
9	(0,0,1,0)	$\{1,5,11\}$	0
10	(0,0,0,0)	$\{1,5,11\}$	5
11	(1,1,1,0)	$\{4,6,11,12\}$	4
12	(1,0,1,0)	$\{4,6,11,12\}$	4
13	(1,0,1,0)	$\{1,2,3,4,5,6,7,8,9,10,11,12\}$	9
14	(1,0,0,1)	$\{9,10,11\}$	9
15	(0,1,0,1)	$\{9,10,11,12\}$	9
16	(0,1,0,1)	$\{4,6,11,12\}$	0
17	(1,1,0,1)	$\{4,6,11,12\}$	0
18	(1,1,1,0)	$\{2,5,8,12\}$	8

Table 5.1: Transaction data in example data set.

5.4.2 Building a customer-level model via recursive partitioning

We now present an approach for building an approximation of customer-level choice model. Before describing the approach, let us describe the model that we will build. In the model that we would like to build, we would like to partition the set of customer attribute vectors $\mathcal{C}$ into finitely many groups or segments of customers:

$$\mathcal{C} = \mathcal{C}^1 \cup \mathcal{C}^2 \cup \cdots \cup \mathcal{C}^K,$$

where $\mathcal{C}^i \cap \mathcal{C}^j = \emptyset$, i.e., the groups are pairwise disjoint and do not overlap.

Each group k corresponds to a choice model $\hat{\mathbb{P}}(\cdot \,|\, \cdot; \mathcal{C}^k)$. For a customer with attribute vector $\mathbf{c}$, that is offered the assortment S, the probability that the customer chooses product i is then given

$$\mathbb{P}(i \,|\, S; \mathbf{c}) = \sum_{k=1}^{K} \mathbb{I}\{\mathbf{c} \in \mathcal{C}^k\} \cdot \hat{\mathbb{P}}(i \,|\, S; \mathcal{C}^k)$$

i.e., we first resolve which group/segment the customer belongs to, and then use the

choice model that corresponds to that segment for making the prediction for the customer at hand.

We now propose an algorithm for building such a model. We start with a single segment $\mathcal{C}^1$ consisting of all customer attribute vectors (i.e., $\mathcal{C}^1 = \mathcal{C}$) and one choice model, $\hat{\mathbb{P}}(\cdot \mid \cdot; \mathcal{C}^1)$ that is estimated from the transactions in this single segment. Then, we consider splitting the segment into two segments – a so-called left-hand segment and a right-hand segment. We consider such a split for every customer attribute $j \in \{1, \ldots, m\}$. For each attribute j, we thus have a left-hand segment $\mathcal{C}^{L,j}$ and a right-hand segment $\mathcal{C}^{R,j}$ that are defined as

$$\mathcal{C}^{L,j} = \{\mathbf{c} \in \mathcal{C}^1 \mid c_j = 0\},$$
$$\mathcal{C}^{R,j} = \{\mathbf{c} \in \mathcal{C}^1 \mid c_j = 1\},$$

i.e., the left-hand segment is just $\mathcal{C}^1$ with the constraint that attribute j is fixed to 0, while the right-hand segment fixes attribute j to 1.

For each candidate split, we fit a choice model to the transactions in the left-hand and the right-hand segment, namely, we fit $\hat{\mathbb{P}}(\cdot \mid \cdot; \mathcal{C}^{L,j})$ and $\hat{\mathbb{P}}(\cdot \mid \cdot; \mathcal{C}^{R,j})$ from $\mathcal{T}^{L,j}$ and $\mathcal{T}^{R,j}$, respectively, where $\mathcal{T}^{L,j}$ is the set of left-hand transactions and $\mathcal{T}^{R,j}$ is the set of right-hand transactions. We also compute the model fit of those segments, denoted by $\mathcal{L}^{L,j}$ and $\mathcal{L}^{R,j}$. By fitting two different models, one to each of $\mathcal{T}^{L,j}$ and $\mathcal{T}^{R,j}$ the overall fit will be at least as good as the fit we achieve if we were constrained to fit the same model to both partitions. We execute the split that results in the best improvement in the model fit relative to the parent segment.

Upon executing the split, the single segment $\mathcal{C}^1$ is replaced by the two new segments. The procedure then repeats repeats anew from these new segments.

The algorithm is described generically as Algorithm 4.

Before continuing on, it is important to remark on several important aspects of this algorithm. First, there are two conditions in the algorithm that are as yet unexplained: on line 3, we only proceed to split a segment $\mathcal{C}'$ if it is "splittable", and on line 10, we build a set J of "acceptable splits". We define these terms as follows:

Algorithm 4 Recursive partitioning algorithm for estimating segmented choice model.

Require: Set of transactions $\{1, \ldots, T\}$; initial segment $\mathcal{C}$; segment collection $\mathcal{P} = \{\mathcal{C}\}$; splittable segments $\mathcal{P}^S = \{\mathcal{C}\}$.

while $\mathcal{P}^S \neq \emptyset$ **do**
  for $\mathcal{C}' \in \mathcal{P}^S$ **do**
    if $\mathcal{C}'$ is splittable **then**
      For each customer attribute j:
        Compute left hand customer segment $\mathcal{C}^{L,j}$, right hand partition $\mathcal{C}^{R,j}$
        Estimate choice model $\hat{\mathbb{P}}(\cdot \,|\, \cdot; \mathcal{C}^{L,j})$ from left-hand transactions, $\mathcal{T}^{L,j}$
        Compute left-hand model fit $\mathcal{L}^{L,j}$
        Estimate choice model $\hat{\mathbb{P}}(\cdot \,|\, \cdot; \mathcal{C}^{R,j})$ from right-hand transactions, $\mathcal{T}^{R,j}$
        Compute right-hand model fit $\mathcal{L}^{R,j}$
      Set $J = \{j \,|\, j \text{ is an acceptable split attribute}\}$
      if $J \neq \emptyset$ **then**
        Set $j^* = \arg\max_{j \in J}(\mathcal{L}^{L,j} + \mathcal{L}^{R,j})$
        Set $\mathcal{P}^S = \mathcal{P}^S \setminus \{\mathcal{C}'\} \cup \{\mathcal{C}^{L,j^*}, \mathcal{C}^{R,j^*}\}$
        Set $\mathcal{P} = \mathcal{P} \setminus \{\mathcal{C}'\} \cup \{\mathcal{C}^{L,j^*}, \mathcal{C}^{R,j^*}\}$
      else
        Set $\mathcal{P}^S = \mathcal{P}^S \setminus \{\mathcal{C}'\}$
      end if
    else
      Set $\mathcal{P}^S = \mathcal{P}^S \setminus \{\mathcal{C}'\}$
    end if
  end for
end while
return Collection of segments $\mathcal{P}$; collection of segment-specific choice models $\{\hat{\mathbb{P}}(\cdot \,|\, \cdot; \mathcal{C}') \,\big|\, \mathcal{C}' \in \mathcal{P}\}$.

- A segment $\mathcal{C}'$ is *splittable* if it contains more than $T_{\min}$ transactions.

- An attribute j is an *acceptable split attribute* if the resulting left and right hand segments each contain more than $T_{\min,split}$ transactions.

The purpose of each of these checks is to ensure that the ultimate predictive model does not overfit the transaction data.

Second, we have so far not described what exactly $\hat{\mathbb{P}}(\cdot \mid \cdot; \mathcal{C}')$ is. The choice of parametric family is open to the modeler. In the numerical results we will consider, we will consider using a simple multinomial logit model:

$$\hat{\mathbb{P}}(i \mid S; \mathcal{C}') = \frac{\exp(u_{i,\mathcal{C}'})}{\sum_{j \in S \cup \{0\}} \exp(u_{j,\mathcal{C}'})},$$

where $u_{i,\mathcal{C}'}$ is the estimated utility of product i for customers in segment $\mathcal{C}'$. For this particular choice of model, the left and right hand fits $\mathcal{L}^{L,j}$ and $\mathcal{L}^{R,j}$ are just the log likelihoods of the MNL models that were estimated from the left hand and right hand transaction sets $\mathcal{T}^{L,j}$ and $\mathcal{T}^{R,j}$ respectively.

5.5 Results

We now present a small numerical study to demonstrate the benefit of the tree-based approach presented in Section 5.4.2. Our main insight is that the tree-based approach can provide a significant improvement in revenue over a uniform strategy based on the MNL model, and the revenue attained by our tree-based approach is close to the revenue attained by a model that knows the ground truth model exactly.

We set up our experiment as follows. For a fixed value of n and M, we first randomly generate the ground truth model. The ground truth model that we will consider is a customer-specific MNL model where the utilities are additive in the customer attributes:

$$\mathbb{P}(i \mid S; \mathbf{c}) = \frac{\exp(\sum_{j=1}^{M} u_{i,j} c_j)}{\sum_{i' \in S \cup \{0\}} \exp(\sum_{j=1}^{M} u_{i',j} c_j)}.$$

Each utility value $u_{i,j}$ is drawn uniformly at random from the interval $[0, 10]$.

We then generate the transaction data: we generate $\{(\mathbf{c}^t, S^t, p^t)\}_{t=1}^T$, i.e., T tuples consisting of the customer attributes, the offered assortment and the choice of the customer. We generate $T = 2000$ such tuples. Each customer vector $\mathbf{c}^t$ is drawn uniformly from $\mathcal{C} = \{0, 1\}^M$. Each set S^t is uniformly randomly selected from 20 different assortments that were uniformly selected from the set of 2^n possible assortments. Each choice p^t is selected according to the probability distribution given by $\mathbb{P}(\cdot \mid S^t; \mathbf{c}^t)$.

Using the data $\{(\mathbf{c}^t, S^t, p^t)\}_{t=1}^T$, we consider two different assortment strategies:

- **Uniform MNL**: In this strategy, we ignore the customer attribute information, and we simply fit an MNL using the whole T offered assortments and choices. In doing so, we obtain a vector of utilities $(u_{0,unif}, u_{1,unif} \ldots, u_{n,unif})$. We find the best optimal assortment for this utility vector:

$$S^* = \arg \max_{S \subseteq \{1,\ldots,n\}} \frac{\sum_{i \in S} r_i \exp(u_{i,unif})}{\sum_{i \in S \cup \{0\}} \exp(u_{i,unif})}.$$

  The uniform strategy then simply involves offering every customer $\mathbf{c}$ the same assortment S^*:

$$S^*_{unif}(\mathbf{c}) = S^*.$$

- **Tree strategy**: In this strategy, we build a collection of segments using Algorithm 4 and a collection of segment-specific MNL models. Letting $\mathcal{P}$ denote the collection of segments and letting $\mathbf{u}_{\mathcal{C}'}$ denote the MNL utility vector of segment $\mathcal{C}' \in \mathcal{P}$, we define $S^*_{\mathcal{C}'}$ as the optimal decision for segment $\mathcal{C}'$:

$$S^*_{\mathcal{C}'} = \arg \max_{S \subseteq \{1,\ldots,n\}} \frac{\sum_{i \in S} r_i \exp(u_{i,\mathcal{C}'})}{\sum_{i \in S \cup \{0\}} \exp(u_{i,\mathcal{C}'})}.$$

  The tree strategy then involves offering every customer $\mathbf{c}$ the assortment corresponding to the segment they are in:

$$S^*_{tree}(\mathbf{c}) = \{S^*_{\mathcal{C}'}, \quad \text{if } \mathbf{c} \in \mathcal{C}'.$$

With regard to the parameters of Algorithm 4, we set both $T_{\min} = 30$ and $T_{\min,split} = 30$.

In addition to these strategies, we also consider the optimal strategy given knowledge the ground truth model: for each $\mathbf{c}$, we compute

$$S^*_{GTO}(\mathbf{c}) = \arg \max_{S \subseteq \{1,\ldots,n\}} \sum_{i \in S} r_i \cdot \mathbb{P}(i \mid S; \mathbf{c}).$$

We refer to this strategy as the **ground truth optimal** (GTO) strategy. (Note that the GTO strategy knows the choice model conditional on the customer attribute vector $\mathbf{c}$, but is not able to perfectly predict the customer's choice given an assortment S and customer attribute vector $\mathbf{c}$. For the ground truth MNL model, the later type of requirement is akin to knowing the random Gumbel errors in the random utility specification of the MNL model, or equivalently, knowing the ranking the customer will use to choose. A strategy that can anticipate the random errors/rankings will lead to higher out-of-sample revenues than one that can only anticipate the precise choice model.)

To evaluate each strategy, we draw $T_{OOS} = 10,000$ customers to test each strategy out-of-sample (OOS). For each $t \in \{1, \ldots, T_{OOS}\}$, we draw a customer attribute vector $\mathbf{c}^{t,OOS}$. For each such customer, we draw the rank list corresponding to the $n+1$ options according to the ground truth model. Mathematically, letting $\sigma^{t,OOS}$ denote the rank list of the out-of-sample customer t, the average out-of-sample revenue $R_{OOS}(S^*)$ of a policy S^* that maps customer attributes to assortments is given by

$$R_{OOS}(S^*) = 1/T_{OOS} \cdot \sum_{t=1}^{T_{OOS}} \sum_{i \in S^*(\mathbf{c}^{t,OOS})} r_i \cdot \mathbb{I}\{i = \arg \min_{i' \in S^*(\mathbf{c}^{t,OOS}) \cup \{0\}} \sigma^{t,OOS}(i')\}.$$

For a strategy S^*, we compute its optimality gap relative to the GTO strategy:

$$G = 100\% \times \frac{R_{OOS}(S^*_{GTO}) - R_{OOS}(S^*)}{R_{OOS}(S^*_{GTO})}.$$

Table 5.2 displays the results for 20 random instances – where each instance is

Inst.	Uniform ($T = 2000$) Rev ($)	Gap (%)	Tree ($T = 500$) Rev ($)	Gap (%)	Tree ($T = 1000$) Rev ($)	Gap (%)	Tree ($T = 2000$) Rev	Gap (%)	GTO Rev ($)
1	53.5	6.32	56.3	1.49	56.4	1.26	57.1	0.05	57.2
2	83.4	4.91	84.3	3.92	80.6	8.17	83.3	5.09	87.7
3	61.1	5.87	64.3	0.92	64.8	0.23	63.5	2.1	64.9
4	48	10.55	52.1	2.76	53.2	0.8	51.8	3.46	53.6
5	33.2	8.52	32.4	10.55	35.2	2.75	33.5	7.51	36.2
6	66.7	14.09	75.1	3.26	73.4	5.5	77.5	0.17	77.6
7	86.1	2.53	88.2	0.15	88.2	0.09	88.1	0.23	88.3
8	62.6	6.61	62	7.58	63.2	5.83	63.3	5.59	67.1
9	55.6	15.1	63.5	2.98	60.7	7.36	64.2	1.92	65.5
10	57.2	23.5	70.1	6.15	69.4	7.1	69.3	7.3	74.7
11	42.5	7.45	40.4	12.07	42.6	7.3	42.6	7.31	46
12	47.6	17.86	53.7	7.37	53.3	8.06	57.9	0.07	57.9
13	76.4	2.57	74.2	5.25	71.1	9.32	67.4	13.99	78.4
14	72.8	1.08	70.8	3.8	70.7	3.93	71.8	2.42	73.6
15	78	10.5	78.3	10.16	81.3	6.72	82.3	5.6	87.2
16	82.6	0.75	82	1.51	83	0.34	83.2	0.02	83.2
17	96.8	0.43	94.8	2.51	96.7	0.5	95	2.32	97.2
18	72.3	9.06	76.2	4.07	77	3.15	78.4	1.4	79.5
19	58.2	3.7	58.8	2.73	59.5	1.44	59.9	0.77	60.4
20	61.4	6.26	63.8	2.65	64.3	1.87	65	0.8	65.5
Avg.	–	7.88	–	4.6	–	4.09	–	3.41	–

Table 5.2: Results for $n = 10$, $M = 5$. ("Rev" indicates the out-of-sample revenue; "Gap" indicates the gap metric G.)

defined by a random ground truth model, random training transaction set and random out-of-sample customer and ranking sets generated as detailed above – with $n = 10$ products and $M = 5$ customer attributes. We compare the uniform MNL strategy that uses all $T = 2000$ transactions to the tree strategy with $T = 500$, $T = 1000$ and $T = 2000$ transactions. We can see from this table that in general, the tree strategy outperforms the uniform MNL strategy and leads to lower average optimality gaps. More precisely, the uniform MNL with all 2000 transactions achieves an average optimality gap of about 7.9% over the 20 instances, while the tree strategies with 500, 1000 and 2000 transactions achieve optimality gaps of 4.6%, 4.1% and 3.4% respectively; in other words, the tree strategy is able to substantially improve on the uniform MNL policy. Note also that the optimality gaps are relatively small; with $T = 2000$ transactions, the tree strategy already achieves about 96.6% of the revenue of the GTO strategy, which is the best that we can hope for in this problem.

5.6 Conclusion

In this chapter, we presented a promising new method for making personalized assortment decisions. The method is based on building a new customer-level choice model through the technique of recursive partitioning that has been successfully used in classification and regression problems, and using the choice probability predictions produced by this model to guide the selection of an assortment. We showed in numerical results with simulated data that this approach outperforms the traditional uniform approach and achieves revenues that are moderately close to the best possible revenue that would be possible if one knew the exact choice model describing each individual customer.

Chapter 6

Conclusions

In this thesis we have studied following question: how do we transform data and models into effective decisions? We have attempted to provide an answer to this question in the following, concrete settings:

1. In Chapter 2, we proposed a new type of linear optimization formulation for a large class of MDPs called decomposable MDPs, and a high quality heuristic derived from this formulation. We showed that the formulation improves on the state of the art both theoretically, in that it provides tighter bounds on the optimal value function than three other ADP formulations, and practically, in that the heuristic policy provides expected rewards that are considerably closer to optimality than alternate ADP approaches.

2. In Chapter 3, we proposed a new approach for making product line decisions under uncertainty. The approach is based on identifying a set of models that could represent the choice behavior of the customer population, and making decisions that optimize against the worst-case expected per-customer revenue taken over this set of models. Using real conjoint data, we showed that traditional product line approaches can lead to poor product lines when the choice model is misspecified, while our approach in contrast can lead to better performance in the presence of uncertainty.

3. In Chapter 4, we proposed a new two-step approach for making assortment

decisions from limited transaction data that involves estimating a ranking-based model of choice, and then finding the assortment that maximizes the expected revenue under this model. We showed through experiments with synthetic data that this type of approach is scalable and outperforms alternative parametric and non-parametric proposals in both out-of-sample predictive accuracy and expected revenue.

4. In Chapter 5, we proposed a new type of segmentation approach for making personalized assortment decisions using data on the choices of previous customers, as well as their personal attributes. Through an experiment with synthetic data, we show that such an approach is able to provide an improvement over a "uniform" approach that offers the same assortment to all customers, and achieves revenues that are moderately close to the best achievable revenue.

Appendix A

Proofs, Counterexamples and Derivations for Chapter 2

A.1 Proofs

A.1.1 Proof of Proposition 1

This result follows directly from observations in [76]. To see this, observe that problem (2.2) can be expressed as a type of doubly infinite linear optimization problem (problem (P) in [76]) where each variable appears in finitely many constraints and each constraint involves only finitely many variables. Note that problem (P) in [76] contains explicit upper bounds on all of the variables, which problem (2.2) does not appear to have; however, note that the variables in problem (2.2), by virtue of the constraints, are upper bounded by 1. In this way, problem (2.2) can be cast as problem (P) from [76]. Assumption A of [76] holds (that the feasible region of problem (2.2) is nonempty; this will be established in the proof of Proposition 2, where we will directly construct a feasible solution to problem (2.2)). Assumption B of [76] also holds (that the objective function is uniformly convergent on the feasible region; this holds due to the discounting in the objective function). With these two assumptions and following the argument in Section 2 of [76], it follows that there exists an optimal solution to problem (2.2). □

A.1.2 Proof of Proposition 2

To prove this, we will construct a feasible solution to problem (2.2) whose objective value in (2.2) is the same as $J^*(\mathbf{s})$. For each $t \in \{1, 2, \dots\}$, set

$$x^m_{ka}(t) = \mathbb{P}(s^m(t) = k,\ \pi^*(\mathbf{s}(t)) = a)$$
$$A_a(t) = \mathbb{P}(\pi^*(\mathbf{s}(t)) = a),$$

where $\{\mathbf{s}(t)\}_{t=1}^{\infty}$ is the stochastic process of the system state that starts in state $\mathbf{s} = (s^1, \dots, s^M)$, operated according to the optimal policy π^* that yields the optimal value function $J^*(\cdot)$. We now verify that the solution $(\mathbf{x}, \mathbf{A})$ is indeed feasible.

For constraint (2.2b), we have for any $m \in \{1, \dots, M\}$, $j \in \mathcal{S}^m$ and $t \in \{2, 3, 4, \dots\}$,

$$\begin{aligned}
\sum_{a \in \mathcal{A}} x^m_{ja}(t) &= \sum_{a \in \mathcal{A}} \mathbb{P}(s^m(t) = j,\ \pi^*(\mathbf{s}(t)) = a) \\
&= \mathbb{P}(s^m(t) = j) \\
&= \sum_{k \in \mathcal{S}^m} \sum_{\tilde{a} \in \mathcal{A}} \mathbb{P}(s^m(t) = j,\ s^m(t-1) = k,\ \pi^*(\mathbf{s}(t-1)) = \tilde{a}) \\
&= \sum_{k \in \mathcal{S}^m} \sum_{\tilde{a} \in \mathcal{A}} \mathbb{P}(s^m(t) = j \mid s^m(t-1) = k,\ \pi^*(\mathbf{s}(t-1)) = \tilde{a}) \\
&\qquad \cdot \mathbb{P}(s^m(t-1) = k,\ \pi^*(\mathbf{s}(t-1)) = \tilde{a}) \\
&= \sum_{k \in \mathcal{S}^m} \sum_{\tilde{a} \in \mathcal{A}} p^m_{kja} \cdot x^m_{ka}(t-1)
\end{aligned}$$

where the first equality follows by the definition of $x^m_{ka}(t)$, the second and third equalities follow by the countable additivity of probability and the law of total probability, the fourth by conditioning on the state and action at $t-1$ and the final step by using the definition of p^m_{kja} and $x^m_{ka}(t)$.

Similarly, for constraint (2.2c), for any $a \in \{1, \dots, M\}$ and $t \in \{1, 2, \dots\}$ we have

$$\sum_{k \in \mathcal{S}^m} x^m_{ka}(t) = \sum_{k \in \mathcal{S}^m} \mathbb{P}(s^m(t) = k,\ \pi^*(\mathbf{s}(t)) = a) = \mathbb{P}(\pi^*(\mathbf{s}(t)) = a) = A_a(t).$$

For constraint (2.2d), our earlier reasoning gives us that for any $m \in \{1, \ldots, M\}$, $k \in \mathcal{S}^m$,

$$\sum_{a=1}^{M} x_{k,a}^m = \sum_{a \in \mathcal{A}} \mathbb{P}(s^m(1) = k,\ \pi^*(\mathbf{s}(1)) = a) = \mathbb{P}(s^m(1) = k) = \alpha_k^m(\mathbf{s}).$$

Lastly, since $\mathbf{x}$ and $\mathbf{A}$ are defined as probabilities, it follows that

$$\begin{aligned} &x_{ka}^m(t) \geq 0, \quad \forall\ m \in \{1, \ldots, M\}, a \in \mathcal{A},\ k \in \mathcal{S}^m,\ t \in \{1, 2, \ldots\}, \\ &A_a(t) \geq 0, \quad \forall\ a \in \mathcal{A},\ t \in \{1, 2, \ldots\}. \end{aligned}$$

This establishes that $(\mathbf{x}, \mathbf{A})$ is a feasible solution to (2.2). We will now verify that its objective value is identical to $J^*(\mathbf{s})$. By the definition of the optimal value function, we have

$$J^*(\mathbf{s}) = \mathbb{E}\left[\sum_{t=1}^{\infty} \sum_{m=1}^{M} \beta^{t-1} \cdot g_{s^m(t), \pi^*(\mathbf{s}(t))}^m\right] = \sum_{t=1}^{\infty} \sum_{m=1}^{M} \beta^{t-1} \cdot \mathbb{E}\left[g_{s^m(t), \pi^*(\mathbf{s}(t))}^m\right],$$

where the second step follows by the monotone convergence theorem (since β and all the g_{ka}^m values are nonnegative) and the linearity of expectation. Observe now that

$$\mathbb{E}[g_{s^m(t), \pi^*(\mathbf{s}(t))}^m] = \sum_{k \in \mathcal{S}^m} \sum_{a \in \mathcal{A}} g_{ka}^m \cdot \mathbb{P}(s^m(t) = k,\ \pi^*(\mathbf{s}(t)) = a) = \sum_{k \in \mathcal{S}^m} \sum_{a \in \mathcal{A}} g_{ka}^m \cdot x_{ka}^m(t).$$

We therefore have

$$J^*(\mathbf{s}) = \sum_{t=1}^{\infty} \sum_{m=1}^{M} \beta^{t-1} \cdot \mathbb{E}\left[g_{s^m(t), \pi^*(\mathbf{s}(t))}^m\right] = \sum_{t=1}^{\infty} \sum_{m=1}^{M} \sum_{k \in \mathcal{S}^m} \sum_{a \in \mathcal{A}} \beta^{t-1} \cdot g_{ka}^m \cdot x_{ka}^m,$$

in other words, the value function evaluated at the starting state $\mathbf{s}$ is exactly the objective value of the $(\mathbf{x}, \mathbf{A})$ solution that we constructed. Since $(\mathbf{x}, \mathbf{A})$ is a feasible solution for (2.2), it follows that $J^*(\mathbf{s}) \leq Z^*(\mathbf{s})$ which is the required result. $\square$

A.1.3 Proof of Proposition 3

Using the same type of reasoning that we used in the proof of Proposition 2 to show that the objective function of the constructed solution was equal to $J^*(\mathbf{s})$, one can show that the objective value of the optimal fluid solution $(\mathbf{x}, \mathbf{A})$ is equal to $\mathbb{E}\left[\sum_{t=1}^{\infty}\sum_{m=1}^{M}\beta^{t-1}\cdot g^m_{s^m(t),\pi(t,\mathbf{s}(t))}\right]$, i.e., the expected discounted reward when the system is operated according to the policy π and the system starts in state $\mathbf{s}$. Since $J^*(\mathbf{s})$ is the maximum over all policies of the expected discounted reward of the system, it follows that $J^*(\mathbf{s})$ is an upper bound on $\mathbb{E}\left[\sum_{t=1}^{\infty}\sum_{m=1}^{M}\beta^{t-1}\cdot g^m_{s^m(t),\pi(t,\mathbf{s}(t))}\right]$, and thus that $J^*(\mathbf{s}) \geq Z^*(\mathbf{s})$. $\square$

A.1.4 Proof of Theorem 1

Let $\mathbf{s} \in \mathcal{S}$, and let $(\mathbf{x}(\mathbf{s}), \mathbf{A}(\mathbf{s}))$ be the solution of the corresponding fluid problem. To prove the theorem, we will show that for state $\mathbf{s}$, any action a with $A_a(1,\mathbf{s}) > 0$ must be greedy with respect to the optimal value function J^*. By standard results in dynamic programming theory (see, e.g., [12]), this is sufficient to prove that π as we have defined it is an optimal policy.

By Propositions 2 and 3, we have that $Z^*(\mathbf{s}) = J^*(\mathbf{s})$. Since $(\mathbf{x}(\mathbf{s}), \mathbf{A}(\mathbf{s}))$ is achievable, let $\bar{\pi}$ be the (possibly nondeterministic, nonstationary) policy that achieves it and let $\{\mathbf{s}(t)\}_{t=1}^{\infty}$ be the stochastic process of the system state when operated according to $\bar{\pi}$. We then have that

$$
\begin{aligned}
J^*(\mathbf{s}) &= Z^*(\mathbf{s}) \\
&= \mathbb{E}\left[\sum_{t=1}^{\infty}\sum_{m=1}^{M}\beta^{t-1}g^m_{s^m(t),\bar{\pi}(t,\mathbf{s}(t))}\right] \\
&= \sum_{m=1}^{M}\mathbb{E}[g^m_{s^m(1),\bar{\pi}(\mathbf{s}(1),1)}] + \mathbb{E}\left[\sum_{t=2}^{\infty}\sum_{m=1}^{M}\beta^{t-1}g^m_{s^m(t),\bar{\pi}(t,\mathbf{s}(t))}\right] \\
&= \sum_{m=1}^{M}\mathbb{E}[g^m_{s^m(1),\bar{\pi}(\mathbf{s}(1),1)}] + \sum_{\tilde{\mathbf{s}}\in\mathcal{S}}\mathbb{E}\left[\sum_{t=2}^{\infty}\sum_{m=1}^{M}\beta^{t-1}g^m_{s^m(t),\bar{\pi}(t,\mathbf{s}(t))}\,\middle|\,\mathbf{s}(2)=\tilde{\mathbf{s}}\right]\cdot\mathbb{P}(\mathbf{s}(2)=\tilde{\mathbf{s}}) \\
&= \sum_{m=1}^{M}\sum_{a\in\mathcal{A}}g^m_{s^m,a}A_a(1,\mathbf{s}) + \beta\cdot\sum_{\tilde{\mathbf{s}}\in\mathcal{S}}\mathbb{E}\left[\sum_{t=2}^{\infty}\sum_{m=1}^{M}\beta^{t-2}g^m_{s^m(t),\bar{\pi}(t,\mathbf{s}(t))}\,\middle|\,\mathbf{s}(2)=\tilde{\mathbf{s}}\right]\cdot\mathbb{P}(\mathbf{s}(2)=\tilde{\mathbf{s}})
\end{aligned}
$$

where the second step follows since $(\mathbf{x}(\mathbf{s}), \mathbf{A}(\mathbf{s}))$ is achieved by $\bar{\pi}$; the third step by breaking up the infinite sum; the fourth step by conditioning on the second state; and the fifth step by using the fact that $A_a(1, \mathbf{s}) = \mathbb{P}(\bar{\pi}(\mathbf{s}(1), 1) = a)$ and factoring out a β. Now observe that we must have

$$\sum_{m=1}^{M} \sum_{a \in \mathcal{A}} g^m_{s^m, a} A_a(1, \mathbf{s}) + \beta \cdot \sum_{\tilde{\mathbf{s}} \in \mathcal{S}} \mathbb{E}\left[\sum_{t=2}^{\infty} \sum_{m=1}^{M} \beta^{t-2} g^m_{s^m(t), \bar{\pi}(t, \mathbf{s}(t))} \,\middle|\, \mathbf{s}(2) = \tilde{\mathbf{s}} \right] \cdot \mathbb{P}(\mathbf{s}(2) = \tilde{\mathbf{s}})$$
$$= \sum_{m=1}^{M} \sum_{a \in \mathcal{A}} g^m_{s^m, a} A_a(1, \mathbf{s}) + \beta \cdot \sum_{\tilde{\mathbf{s}} \in \mathcal{S}} J^*(\tilde{\mathbf{s}}) \cdot \mathbb{P}(\mathbf{s}(2) = \tilde{\mathbf{s}}). \tag{A.1}$$

This follows because the left hand side is less than or equal to the right hand side, which is true because

$$\mathbb{E}\left[\sum_{t=2}^{\infty} \sum_{m=1}^{M} \beta^{t-2} g^m_{s^m(t), \bar{\pi}(t, \mathbf{s}(t))} \,\middle|\, \mathbf{s}(2) = \tilde{\mathbf{s}} \right] \leq J^*(\tilde{\mathbf{s}})$$

(here, the left hand side is the value of running policy $\bar{\pi}$ from time 2 on at state $\tilde{\mathbf{s}}$, which is clearly at most $J^*(\tilde{\mathbf{s}})$). Note also that the left hand side in equation (A.1) cannot be *strictly* less than the right hand side, because the right hand side value is achieved by a policy (follow $\bar{\pi}$ at $t = 1$, then follow any optimal policy from $t = 2$ on); by our earlier manipulation, the left hand side in equation (A.1) is equal to $J^*(\mathbf{s})$, so a strict inequality would imply a policy that achieves a better value than the optimal value. This is not possible, so equation (A.1) must hold.

Thus, so far, we have that

$$J^*(\mathbf{s}) = \sum_{m=1}^{M} \sum_{a \in \mathcal{A}} g^m_{s^m, a} A_a(1, \mathbf{s}) + \beta \cdot \sum_{\tilde{\mathbf{s}} \in \mathcal{S}} J^*(\tilde{\mathbf{s}}) \cdot \mathbb{P}(\mathbf{s}(2) = \tilde{\mathbf{s}}). \tag{A.2}$$

Since the components are independent, we can write $\mathbb{P}(\mathbf{s}(2) = \tilde{\mathbf{s}})$ as

$$\mathbb{P}(\mathbf{s}(2) = \tilde{\mathbf{s}}) = \sum_{a \in \mathcal{A}} A_a(1, \mathbf{s}) \cdot \prod_{m=1}^{M} p^m_{s^m \tilde{s}^m a}$$

and simplify equation (A.2) as follows:

$$J^*(\mathbf{s}) = \sum_{m=1}^{M} \sum_{a \in \mathcal{A}} g^m_{s^m,a} A_a(1,\mathbf{s}) + \beta \cdot \sum_{\tilde{\mathbf{s}} \in \mathcal{S}} J^*(\tilde{\mathbf{s}}) \cdot \left(\sum_{a \in \mathcal{A}} A_a(1,\mathbf{s}) \cdot \prod_{m=1}^{M} p^m_{s^m \tilde{s}^m a} \right)$$
$$= \sum_{a \in \mathcal{A}} A_a(1,\mathbf{s}) \cdot \left(\sum_{m=1}^{M} g^m_{s^m(1),a} + \beta \cdot \sum_{\tilde{\mathbf{s}} \in \mathcal{S}} J^*(\tilde{\mathbf{s}}) \cdot \prod_{m=1}^{M} p^m_{s^m \tilde{s}^m a} \right). \quad \text{(A.3)}$$

Now recall that $J^*(\mathbf{s})$ is the optimal value function, so it must satisfy the Bellman equation:

$$J^*(\mathbf{s}) = \max_{a \in \mathcal{A}} \left(\sum_{m=1}^{M} g^m_{s^m,a} + \beta \cdot \sum_{\tilde{\mathbf{s}} \in \mathcal{S}} J^*(\tilde{\mathbf{s}}) \cdot \prod_{m=1}^{M} p^m_{s^m \tilde{s}^m a} \right).$$

Given these two expressions, it must be that for any $\bar{a}$ with $A_{\bar{a}}(1,\mathbf{s}) > 0$, we have

$$\sum_{m=1}^{M} g^m_{s^m(1),\bar{a}} + \beta \cdot \sum_{\tilde{\mathbf{s}} \in \mathcal{S}} J^*(\tilde{\mathbf{s}}) \cdot \prod_{m=1}^{M} p^m_{s^m \tilde{s}^m \bar{a}} = \max_{a \in \mathcal{A}} \left(\sum_{m=1}^{M} g^m_{s^m,a} + \beta \cdot \sum_{\tilde{\mathbf{s}} \in \mathcal{S}} J^*(\tilde{\mathbf{s}}) \cdot \prod_{m=1}^{M} p^m_{s^m \tilde{s}^m a} \right);$$

i.e., that $\bar{a}$ is an action that is greedy with respect to the optimal value function J^*. This must be true; by the definition of the maximum, the right hand side is clearly greater than or equal to the left (this is true for any $\bar{a} \in \mathcal{A}$, not only those with $A_{\bar{a}}(1,\mathbf{s}) > 0$). It cannot be strictly greater, because then equation (A.3) could not hold (the right hand side in (A.3) would have to be strictly less than $J^*(\mathbf{s})$, as the $A_a(1,\mathbf{s})$ values are nonnegative and sum to one).

Finally, note that the action $a = \arg\max_{\bar{a} \in \mathcal{A}} A_{\bar{a}}(1,\mathbf{s})$ clearly satisfies $A_a(1,\mathbf{s}) > 0$, since the $A_a(t)$ variables are nonnegative and have unit sum. Thus, the action $a = \arg\max_{\bar{a} \in \mathcal{A}} A_{\bar{a}}(1,\mathbf{s})$ must be greedy with respect to the optimal value function. This establishes that the policy π, as defined in the statement of Theorem 1, must be an optimal policy, which is the required result. □

A.1.5 Proof of Proposition 4

Proof of Part (a)

To prove (a), we will define solution to problem (2.3) that (i) is feasible for problem (2.3) and (ii) achieves an objective value in (2.3) that is equal to $J^*(\mathbf{s})$. Since $Z_T^*(\mathbf{s})$ is the optimal value of problem (2.3) and our constructed feasible solution achieves a value of $J^*(\mathbf{s})$, it will follow that $J^*(\mathbf{s}) \leq Z_T^*(\mathbf{s})$.

Let π^* be the optimal policy that solves problem (2.1) and let us consider the following solution for problem (2.3):

$$
\begin{aligned}
&x_{ka}^m(t) = \mathbb{P}(s^m(t) = k,\ \pi^*(\mathbf{s}(t)) = a), \quad \forall\ t \in \{1, \ldots, T\},\\
&A_a(t) = \mathbb{P}(\pi^*(\mathbf{s}(t)) = a), \quad \forall\ t \in \{1, \ldots, T\},\\
&x_{ka}^m(T+1) = \sum_{t=T+1}^{\infty} \beta^{t-(T+1)} \cdot \mathbb{P}(s^m(t) = k,\ \pi^*(\mathbf{s}(t)) = a),\\
&A_a(T+1) = \sum_{t=T+1}^{\infty} \beta^{t-(T+1)} \cdot \mathbb{P}(\pi^*(\mathbf{s}(t)) = a),
\end{aligned}
$$

where $\{\mathbf{s}(t)\}_{t=1}^{\infty}$ is the stochastic process of the complete system that begins in state $\mathbf{s}$ at $t = 1$ and is operated according to policy π^*.

We first need to show that the proposed $(\mathbf{x}, \mathbf{A})$ is feasible. The same steps used in the proof of Proposition 2 can be used to verify that constraints (2.3b), (2.3d) (for $t \in \{1, \ldots, T\}$) and (2.3e) hold. To verify that constraint (2.3c) is satisfied, we apply similar reasoning as for constraint (2.3b), but in a long-run discounted way:

$$
\begin{aligned}
\sum_{a \in \mathcal{A}} x_{ja}^m(T+1) &= \sum_{a \in \mathcal{A}} \sum_{t=T+1}^{\infty} \beta^{t-(T+1)} \cdot \mathbb{P}(s^m(t) = j,\ \pi^*(\mathbf{s}(t)) = a)\\
&= \sum_{t=T+1}^{\infty} \beta^{t-(T+1)} \sum_{k \in \mathcal{S}^m} \sum_{a \in \mathcal{A}} \mathbb{P}(s^m(t) = j \mid s^m(t-1) = k,\ \pi^*(\mathbf{s}(t)) = a)\\
&\qquad \cdot \mathbb{P}(s^m(t-1) = k,\ \pi^*(\mathbf{s}(t)) = a)\\
&= \sum_{t=T+1}^{\infty} \beta^{t-(T+1)} \sum_{k \in \mathcal{S}^m} \sum_{a \in \mathcal{A}} p_{kja}^m \cdot \mathbb{P}(s^m(t-1) = k,\ \pi^*(\mathbf{s}(t-1)) = a)
\end{aligned}
$$

$$
\begin{aligned}
&= \sum_{k\in\mathcal{S}^m}\sum_{a\in\mathcal{A}} p^m_{kja}\cdot\mathbb{P}(s^m(T)=k,\ \pi^*(\mathbf{s}(T))=a)\\
&\quad+\beta\cdot\sum_{t=T+2}^{\infty}\beta^{t-(T+2)}\sum_{k\in\mathcal{S}^m}\sum_{a\in\mathcal{A}} p^m_{kja}\cdot\mathbb{P}(s^m(t-1)=k,\ \pi^*(\mathbf{s}(t-1))=a)\\
&= \sum_{k\in\mathcal{S}^m}\sum_{a\in\mathcal{A}} p^m_{kja}\cdot\mathbb{P}(s^m(T)=k,\ \pi^*(\mathbf{s}(T))=a)\\
&\quad+\beta\cdot\sum_{k\in\mathcal{S}^m}\sum_{a\in\mathcal{A}} p^m_{kja}\left[\sum_{t=T+1}^{\infty}\beta^{t-(T+1)}\cdot\mathbb{P}(s^m(t)=k,\ \pi^*(\mathbf{s}(t))=a)\right]\\
&= \sum_{k\in\mathcal{S}^m}\sum_{a\in\mathcal{A}} p^m_{kja}\cdot x^m_{ka}(T)+\beta\cdot\sum_{k\in\mathcal{S}^m}\sum_{a\in\mathcal{A}} p^m_{kja}x^m_{ka}(T+1).
\end{aligned}
$$

To verify constraint (2.3d) for $t=T+1$, observe that

$$
\begin{aligned}
\sum_{k\in\mathcal{S}^m} x^m_{ka}(T+1) &= \sum_{k\in\mathcal{S}^m}\sum_{t=T+1}^{\infty}\beta^{t-(T+1)}\cdot\mathbb{P}(s^m(t)=k,\ \pi^*(\mathbf{s}(t))=a)\\
&= \sum_{t=T+1}^{\infty}\beta^{t-(T+1)}\sum_{k\in\mathcal{S}^m}\mathbb{P}(s^m(t)=k,\ \pi^*(\mathbf{s}(t))=a)\\
&= \sum_{t=T+1}^{\infty}\beta^{t-(T+1)}\cdot\mathbb{P}(\pi^*(\mathbf{s}(t))=a)\\
&= A_a(T+1).
\end{aligned}
$$

Lastly, by our construction of $\mathbf{x}$ and $\mathbf{A}$, it is clear that both are nonnegative, satisfying the non-negativity constraints of problem (2.3).

Now, let us consider the objective value of $(\mathbf{x},\mathbf{A})$; we have

$$
\begin{aligned}
\mathbb{E}\left[\sum_{t=1}^{\infty}\sum_{m=1}^{M}\beta^{t-1}g^m_{s^m(t),\pi^*(\mathbf{s}(t))}\right] &= \mathbb{E}\left[\sum_{t=1}^{T}\sum_{m=1}^{M}\beta^{t-1}g^m_{s^m(t),\pi^*(\mathbf{s}(t))}\right]\\
&\quad+\beta^{T+1}\cdot\mathbb{E}\left[\sum_{t=T+1}^{\infty}\sum_{m=1}^{M}\beta^{t-(T+1)}g^m_{s^m(t),\pi^*(\mathbf{s}(t))}\right]\\
&= \sum_{t=1}^{T}\sum_{m=1}^{M}\mathbb{E}\left[g^m_{s^m(t),\pi^*(\mathbf{s}(t))}\right]\\
&\quad+\beta^{T+1}\sum_{t=T+1}^{\infty}\sum_{m=1}^{M}\beta^{t-(T+1)}\mathbb{E}\left[g^m_{s^m(t),\pi^*(\mathbf{s}(t))}\right]
\end{aligned}
$$

$$
\begin{aligned}
&= \sum_{t=1}^{T} \sum_{m=1}^{M} \beta^{t-1} \left(\sum_{k \in \mathcal{S}^m} \sum_{a \in \mathcal{A}} g_{ka}^m \cdot \mathbb{P}(s^m(t) = k,\ \pi^*(\mathbf{s}(t)) = a) \right) \\
&\quad + \beta^{T+1} \cdot \\
&\quad \sum_{t=T+1}^{\infty} \beta^{t-(T+1)} \left(\sum_{k \in \mathcal{S}^m} \sum_{a \in \mathcal{A}} g_{ka}^m \cdot \mathbb{P}(s^m(t) = k,\ \pi^*(\mathbf{s}(t)) = a) \right) \\
&= \sum_{t=1}^{T} \sum_{m=1}^{M} \sum_{k \in \mathcal{S}^m} \sum_{a \in \mathcal{A}} \beta^{t-1} g_{ka}^m x_{ka}^m(t) \\
&\quad + \beta^{T+1} \cdot \\
&\quad \sum_{k \in \mathcal{S}^m} \sum_{a \in \mathcal{A}} g_{ka}^m \left(\sum_{t=T+1}^{\infty} \beta^{t-(T+1)} \cdot \mathbb{P}(s^m(t) = k,\ \pi^*(\mathbf{s}(t)) = a) \right) \\
&= \sum_{t=1}^{T+1} \sum_{m=1}^{M} \sum_{k \in \mathcal{S}^m} \sum_{a \in \mathcal{A}} \beta^{t-1} g_{ka}^m x_{ka}^m(t)
\end{aligned}
$$

i.e., the objective value of the $(\mathbf{x}, \mathbf{A})$ solution is identical to the optimal expected discounted value $\mathbb{E}[\sum_{t=1}^{\infty} \sum_{m=1}^{M} \beta^{t-1} g_{s^m(t), \pi^*(\mathbf{s}(t))}^m]$, which is exactly $J^*(\mathbf{s})$. Since $(\mathbf{x}, \mathbf{A})$ is feasible, it follows that this objective value is less than $Z_T^*(\mathbf{s})$, which establishes that $J^*(\mathbf{s}) \leq Z_T^*(\mathbf{s})$ and concludes the proof of part (a) of the Theorem. $\square$

Proof of Part (b)

To prove (b), we will use the optimal solution of problem (2.3) with a horizon of $T+1$ to construct a solution for problem (2.3) with a horizon of T that (i) is feasible for problem (2.3) with T and (ii) achieves an objective value of $Z_{T+1}^*(\mathbf{s})$ in problem (2.3). As in part (a) of the theorem, it will then follow that $Z_T^*(\mathbf{s}) \geq Z_{T+1}^*(\mathbf{s})$.

Let $(\bar{\mathbf{x}}, \bar{\mathbf{A}})$ be the optimal solution of problem (2.3) with horizon $T+1$. Define the solution $(\mathbf{x}, \mathbf{A})$ to problem (2.3) with horizon T as follows:

$$
\begin{aligned}
&x_{ka}^m(t) = \bar{x}_{ka}^m(t), && \forall t \in \{1, \ldots, T\}, \\
&x_{ka}^m(T+1) = \bar{x}_{ka}^m(T+1) + \beta \cdot \bar{x}_{ka}^m(T+2), && \\
&A_a(t) = \bar{A}_a(t), && \forall t \in \{1, \ldots, T\}, \\
&A_a(T+1) = \bar{A}_a(T+1) + \beta \cdot \bar{A}_a(T+2). &&
\end{aligned}
$$

By construction, $(\mathbf{x}, \mathbf{A})$ satisfy constraints (2.3b), (2.3e), (2.3f) and (2.3g). The solution $(\mathbf{x}, \mathbf{A})$ also satisfies constraint (2.3d) for $t = 1$ to $t = T$. To show that constraint (2.3c) holds, observe that

$$\begin{aligned}
\sum_{a\in\mathcal{A}} x^m_{ja}(T+1) &= \sum_{a\in\mathcal{A}} \bar{x}^m_{ja}(T+1) + \beta \sum_{a\in\mathcal{A}} \bar{x}^m_{ja}(T+2) \\
&= \sum_{a\in\mathcal{A}} \sum_{k\in\mathcal{S}^m} p^m_{kja} \bar{x}^m_{ka}(T) \\
&\quad + \beta \left(\sum_{k\in\mathcal{S}^m} \sum_{a\in\mathcal{S}^m} p^m_{kja} \bar{x}^m_{ka}(T+1) + \beta \sum_{k\in\mathcal{S}^m} \sum_{a\in\mathcal{A}} p^m_{kja} \bar{x}^m_{ka}(T+2) \right) \\
&= \sum_{a\in\mathcal{A}} \sum_{k\in\mathcal{S}^m} p^m_{kja} \bar{x}^m_{ka}(T) + \beta \sum_{k\in\mathcal{S}^m} \sum_{a\in\mathcal{S}^m} p^m_{kja} \left(\bar{x}^m_{ka}(T+1) + \beta \bar{x}^m_{ka}(T+2) \right) \\
&= \sum_{a\in\mathcal{A}} \sum_{k\in\mathcal{S}^m} p^m_{kja} x^m_{ka}(T) + \beta \sum_{k\in\mathcal{S}^m} \sum_{a\in\mathcal{S}^m} p^m_{kja} x^m_{ka}(T+1).
\end{aligned}$$

To show that constraint (2.3d) for $t = T + 1$, we have that

$$\begin{aligned}
\sum_{k\in\mathcal{S}^m} x^m_{ka}(T+1) &= \sum_{k\in\mathcal{S}^m} \bar{x}^m_{ka}(T+1) + \beta \sum_{k\in\mathcal{S}^m} \bar{x}^m_{ka}(T+2) \\
&= \bar{A}_a(T+1) + \beta \bar{A}_a(T+2) \\
&= A_a(T+1).
\end{aligned}$$

Now, consider the objective of $(\mathbf{x}, \mathbf{A})$:

$$\begin{aligned}
\sum_{t=1}^{T+1} \sum_{m=1}^{M} \sum_{k\in\mathcal{S}^m} \sum_{a\in\mathcal{A}} \beta^{t-1} \cdot g^m_{ka} \cdot x^m_{ka}(t) &= \sum_{t=1}^{T} \sum_{m=1}^{M} \sum_{k\in\mathcal{S}^m} \sum_{a\in\mathcal{A}} \beta^{t-1} \cdot g^m_{ka} \cdot \bar{x}^m_{ka}(t) \\
&\quad + \sum_{m=1}^{M} \sum_{k\in\mathcal{S}^m} \sum_{a\in\mathcal{A}} \beta^{T} \cdot g^m_{ka} \bar{x}^m_{ka}(T+1) \\
&\quad + \sum_{m=1}^{M} \sum_{k\in\mathcal{S}^m} \sum_{a\in\mathcal{A}} \beta^{T+1} g^m_{ka} \bar{x}^m_{ka}(T+2) \\
&= \sum_{t=1}^{T+2} \sum_{m=1}^{M} \sum_{k\in\mathcal{S}^m} \sum_{a\in\mathcal{A}} \beta^{t-1} g^m_{ka} \bar{x}^m_{ka}(t)
\end{aligned}$$

which, by definition of $(\bar{\mathbf{x}}, \bar{\mathbf{A}})$, is exactly $Z^*_{T+1}(\mathbf{s})$. Since $(\mathbf{x}, \mathbf{A})$ is a feasible solution

for problem (2.3) with horizon T, it follows that $Z_T^*(\mathbf{s}) \geq Z_{T+1}^*(\mathbf{s})$, which concludes the proof of (b). $\square$

A.1.6 Proof of Theorem 2

Proof of Part (a)

To prove the result, we will show that $Z_{ALO}^*(\mathbf{s}) \leq Z_{ALR}^*(\mathbf{s})$ and $Z_{ALO}^*(\mathbf{s}) \geq Z_{ALR}^*(\mathbf{s})$. The inequality $Z_{ALO}^*(\mathbf{s}) \leq Z_{ALR}^*(\mathbf{s})$ can be established by applying Corollary 1 of [2], where the weakly coupled MDP is the one defined via the transformation in Section 2.4.3. We thus restrict our focus to the inequality $Z_{ALO}^*(\mathbf{s}) \geq Z_{ALR}^*(\mathbf{s})$.

Before proceeding with the proof, let us transform problem (2.12) (the dual of the Lagrangian relaxation problem (2.11)) for initial state $\mathbf{s}$ to the following equivalent problem:

$$\underset{\mathbf{z},\mathbf{A}}{\text{maximize}} \quad \sum_{m=1}^{M} \sum_{k \in \mathcal{S}^m} \sum_{a \in \mathcal{A}} g_{ka}^m z_{ka}^m \tag{A.4a}$$

$$\text{subject to} \quad \sum_{a \in \mathcal{A}} z_{ja}^m = \alpha_k^m(\mathbf{s}) + \beta \cdot \sum_{k \in \mathcal{S}^m} \sum_{a \in \mathcal{A}} p_{kja}^m z_{ka}^m, \quad \forall\ m \in \{1, \ldots, M\},\ j \in \mathcal{S}^m, \tag{A.4b}$$

$$\sum_{k \in \mathcal{S}^m} z_{ka}^m = A_a, \quad \forall\ m \in \{1, \ldots, M\},\ k \in \mathcal{S}^m, \tag{A.4c}$$

$$z_{ka}^m \geq 0, \quad \forall\ m \in \{1, \ldots, M\},\ k \in \mathcal{S}^m,\ a \in \mathcal{A}, \tag{A.4d}$$

$$A_a \geq 0, \quad \forall\ a \in \mathcal{A}, \tag{A.4e}$$

where A_a captures the expected discounted frequency with which action a is taken from $t = 1$ on. It is straightforward to see that problem (A.4) and (2.12) are equivalent; we have merely reformulated constraint (2.12c) in problem (2.12) through the

A_a variables. The dual of problem (A.4) is

$$\underset{\mathbf{V},\boldsymbol{\phi}}{\text{minimize}} \quad \sum_{m=1}^{M} \sum_{k \in \mathcal{S}^m} \alpha_k^m(\mathbf{s}) V_k^m \tag{A.5a}$$

$$\text{subject to} \quad V_k^m \geq g_{ka}^m + \phi_a^m + \beta \cdot \sum_{j \in \mathcal{S}^m} p_{kja}^m V_j^m, \quad \forall\ m \in \{1, \ldots, M\},\ k \in \mathcal{S}^m,\ a \in \mathcal{A}, \tag{A.5b}$$

$$\sum_{m=1}^{M} \phi_a^m \geq 0, \quad \forall\ a \in \mathcal{A}. \tag{A.5c}$$

Let us now return to proving that $Z_{ALO}^*(\mathbf{s}) \geq Z_{ALR}^*(\mathbf{s})$. Let $\mathbf{J}$ be an optimal solution of problem (2.5). Consider the following solution to problem (A.5):

$$V_k^m = J_k^m, \quad \forall\ m \in \{1, \ldots, M\},\ k \in \mathcal{S}^m,\ a \in \mathcal{A},$$
$$\phi_a^m = \min_{k' \in \mathcal{S}^m} \left\{ J_{k'}^m - g_{k'a}^m - \beta \cdot \sum_{j \in \mathcal{S}^m} p_{k'ja}^m J_j^m \right\}$$

It is straightforward to see that the proposed solution $(\mathbf{V}, \boldsymbol{\phi})$ attains the objective value $Z_{ALO}^*(\mathbf{s})$ in problem (A.5); therefore, it only remains to prove that $(\mathbf{V}, \boldsymbol{\phi})$ is feasible for problem (A.5).

We will first verify constraint (A.5b). Observe that by definition of ϕ_a^m, it satisfies

$$\phi_a^m \leq J_k^m - g_{ka}^m - \beta \cdot \sum_{j \in \mathcal{S}^m} p_{kja}^m J_j^m,$$

for any $m \in \{1, \ldots, M\}$ and $k \in \mathcal{S}^m$, which we can re-arrange to obtain

$$J_k^m \geq g_{ka}^m + \phi_a^m + \beta \cdot \sum_{j \in \mathcal{S}^m} p_{kja}^m J_j^m.$$

This last expression, by our definition of $V_k^m = J_k^m$, is equivalent to

$$V_k^m \geq g_{ka}^m + \phi_a^m + \beta \cdot \sum_{j \in \mathcal{S}^m} p_{kja}^m V_j^m;$$

in other words, the proposed solution $(\mathbf{V}, \boldsymbol{\phi})$ satisfies constraint (A.5b).

Next, let us verify constraint (A.5c). For a given $a \in \mathcal{A}$, we have that

$$\begin{aligned}\sum_{m=1}^{M} \phi_a^m &= \sum_{m=1}^{M} \min_{k' \in \mathcal{S}^m} \left\{ J_{k'}^m - g_{k'a}^m - \beta \cdot \sum_{j \in \mathcal{S}^m} p_{k'ja}^m J_j^m \right\} \\ &= \min_{\bar{\mathbf{s}} \in \mathcal{S}} \left\{ \sum_{m=1}^{M} \left(J_{\bar{s}^m}^m - g_{\bar{s}^m a}^m - \beta \cdot \sum_{j \in \mathcal{S}^m} p_{\bar{s}^m ja}^m J_j^m \right) \right\}\end{aligned}$$

where the first step follows by definition of $\boldsymbol{\phi}$; and the second step, which is the most crucial, follows by the fact that the per-component minimizations are independent of each other and taken over each component's state space, allowing them to be merged together into a single minimization over the system state space.

Now, recall that $\mathbf{J}$ is a feasible solution to problem (2.5) and as such, it satisfies

$$\sum_{m=1}^{M} J_{\bar{s}^m}^m - \sum_{m=1}^{M} g_{\bar{s}^m a}^m - \beta \sum_{m=1}^{M} \sum_{j \in \mathcal{S}^m} p_{\bar{s}^m ja}^m J_j^m \geq 0,$$

for any state $\bar{\mathbf{s}}$ and the action a. Therefore, it follows that

$$\sum_{m=1}^{M} \phi_a^m = \min_{\bar{\mathbf{s}} \in \mathcal{S}} \left\{ \sum_{m=1}^{M} J_{\bar{s}^m}^m - \sum_{m=1}^{M} g_{\bar{s}^m a}^m - \beta \cdot \sum_{m=1}^{M} \sum_{j \in \mathcal{S}^m} p_{\bar{s}^m ja}^m J_j^m \right\} \geq 0$$

which establishes that $(\mathbf{J}, \boldsymbol{\phi})$ satisfies constraint (A.5c). It now follows that $(\mathbf{J}, \boldsymbol{\phi})$ is feasible for problem (2.5) and thus, that $Z_{ALO}^*(\mathbf{s}) \geq Z_{ALR}^*(\mathbf{s})$. Since we have established both $Z_{ALO}^*(\mathbf{s}) \geq Z_{ALR}^*(\mathbf{s})$ and $Z_{ALO}^*(\mathbf{s}) \leq Z_{ALR}^*(\mathbf{s})$, it follows that $Z_{ALO}^*(\mathbf{s}) = Z_{ALR}^*(\mathbf{s})$, which concludes the proof. $\square$

Proof of Part (b)

In the proof of part (a), we essentially showed that any optimal solution to problem (A.5) – the dual of problem (A.4) (which is the transformation of problem (2.12)) – can be used to construct an optimal solution to the ALO problem (2.5), and vice versa. It only remains to show that an optimal solution of the ALR problem (2.11) can be used to construct an optimal solution to problem (A.5); this is straightforward

and is omitted. □

A.1.7 Proof of Theorem 3

Proof of Part (a)

To prove (a), we will use the optimal solution of problem (2.3) with a horizon of T to construct a solution for problem (2.12) that is feasible for problem (2.12) and achieves an objective value of $Z_T^*(\mathbf{s})$ in problem (2.12). It will then follow that $Z_T^*(\mathbf{s}) \leq Z_{ALR}^*(\mathbf{s})$.

Let $(\mathbf{x}, \mathbf{A})$ be the optimal solution of problem (2.3) with a horizon of T. Consider a solution to problem (2.12) defined as follows:

$$z_{ka}^m = \sum_{t=1}^{T+1} \beta^{t-1} \cdot x_{ka}^m(t).$$

To verify that constraint (2.12b) holds, observe that

$$\begin{aligned}
\sum_{a\in\mathcal{A}} z_{ja}^m &= \sum_{a\in\mathcal{A}} \sum_{t=1}^{T+1} \beta^{t-1} \cdot x_{ja}^m(t) \\
&= \sum_{a\in\mathcal{A}} x_{ja}^m(1) + \sum_{t=2}^{T} \beta^{t-1} \cdot \sum_{a\in\mathcal{A}} x_{ja}^m(t) + \beta^T \cdot \sum_{a\in\mathcal{A}} x_{ja}^m(T+1) \\
&= \alpha_j^m(\mathbf{s}) + \sum_{t=2}^{T} \beta^{t-1} \cdot \sum_{k\in\mathcal{S}^m} \sum_{a\in\mathcal{A}} p_{kja}^m x_{ka}^m(t-1) \\
&\quad + \beta^T \left(\sum_{k\in\mathcal{S}^m} \sum_{a\in\mathcal{A}} p_{kja}^m x_{ka}^m(T) + \beta \sum_{k\in\mathcal{S}^m} \sum_{a\in\mathcal{A}} p_{kja}^m x_{ka}^m(T+1) \right) \\
&= \alpha_j^m(\mathbf{s}) + \sum_{k\in\mathcal{S}^m} \sum_{a\in\mathcal{A}} p_{kja}^m \cdot \left(\sum_{t=2}^{T+2} \beta^{t-1} x_{ka}^m(t-1) \right) \\
&= \alpha_j^m(\mathbf{s}) + \beta \sum_{k\in\mathcal{S}^m} \sum_{a\in\mathcal{A}} p_{kja}^m \cdot \left(\sum_{t=1}^{T+1} \beta^{t-1} x_{ka}^m(t) \right) \\
&= \alpha_j^m(\mathbf{s}) + \beta \sum_{k\in\mathcal{S}^m} \sum_{a\in\mathcal{A}} p_{kja}^m \cdot z_{ka}^m.
\end{aligned}$$

To verify that constraint (2.12c) holds, observe that

$$\sum_{k\in\mathcal{S}^m} z_{ka}^m = \sum_{t=1}^{T+1}\sum_{k\in\mathcal{S}^m} \beta^{t-1}x_{ka}^m(t) = \sum_{t=1}^{T+1} A_a(t) = \sum_{t=1}^{T+1}\sum_{k\in\mathcal{S}^{m+1}} \beta^{t-1}x_{ka}^{m+1}(t) = \sum_{k\in\mathcal{S}^{m+1}} z_{ka}^{m+1}.$$

Finally, if we consider the objective of $\mathbf{z}$ in problem (2.12), we see that

$$\sum_{m=1}^{M}\sum_{k\in\mathcal{S}^m}\sum_{a\in\mathcal{A}} g_{ka}^m z_{ka}^m = \sum_{m=1}^{M}\sum_{k\in\mathcal{S}^m}\sum_{a\in\mathcal{A}} g_{ka}^m \beta^{t-1}x_{ka}^m(t) = \sum_{t=1}^{T}\sum_{m=1}^{M}\sum_{k\in\mathcal{S}^m}\sum_{a\in\mathcal{A}} \beta^{t-1}g_{ka}^m x_{ka}^m(t),$$

which is exactly the objective of $(\mathbf{x}, \mathbf{A})$ in problem (2.3) with a horizon of T, and is equal to $Z_T^*(\mathbf{s})$. Since $\mathbf{z}$ is a feasible solution for problem (2.12) and $Z_{ALR}^*(\mathbf{s})$ is the optimal reward, it must be that $Z_{ALR}^*(\mathbf{s}) \geq Z_T^*(\mathbf{s})$, which concludes the proof of part (a). □

Proof of Part (b)

By Theorem 2, we know that $Z_{ALO}^*(\mathbf{s}) = Z_{ALR}^*(\mathbf{s})$ for all $\mathbf{s} \in \mathcal{S}$, and by part (c) of Theorem 3, we know that $Z_T^*(\mathbf{s}) \leq Z_{ALR}^*(\mathbf{s})$. Therefore, it follows that $Z_T^*(\mathbf{s}) \leq Z_{ALO}^*(\mathbf{s})$ for every $\mathbf{s} \in \mathcal{S}$. □

Proof of Part (c)

Part (b) asserts that $Z_T^*(\mathbf{s}) \leq Z_{ALO}^*(\mathbf{s})$, while Proposition 7 (Corollary 1 of [2]) asserts that $Z_{ALO}^*(\mathbf{s}) \leq Z_{CLR}^*(\mathbf{s})$. Combining these two inequalities, we obtain the desired result. □

A.1.8 Proof of Theorem 4

To show that $Z_T^*(\mathbf{s}) = Z_{ALR(T)}^*(\mathbf{s})$, write the dual of the ALR(T) problem (2.13):

$$\begin{aligned}
\underset{\mathbf{z}}{\text{maximize}} \quad & \sum_{t=1}^{T+1}\sum_{m=1}^{M}\sum_{k\in\mathcal{S}^m}\sum_{a\in\mathcal{A}} \beta^{t-1}\cdot g_{ka}^m \cdot z_{ka}^m(t) && \text{(A.6a)}\\
\text{subject to} \quad & \sum_{k\in\mathcal{S}^m}\sum_{a\in\mathcal{A}} p_{kja}^m z_{ka}^m(t) = \sum_{a\in\mathcal{A}} z_{ja}^m(t+1),
\end{aligned}$$

$$\forall\ m \in \{1,\dots,M\},\ j \in \mathcal{S}^m,\ t \in \{1,\dots,T-1\}, \tag{A.6b}$$

$$\sum_{k\in\mathcal{S}^m}\sum_{a\in\mathcal{A}} p^m_{kja} z^m_{ka}(T) + \beta \sum_{k\in\mathcal{S}^m}\sum_{a\in\mathcal{A}} p^m_{kja} z^m_{ka}(T+1) = \sum_{a\in\mathcal{A}} z^m_{ja}(T+1),$$

$$\forall\ m \in \{1,\dots,M\}, j \in \mathcal{S}^m, \tag{A.6c}$$

$$\sum_{a\in\mathcal{A}} z^m_{ka}(1) = \alpha^m_k(\mathbf{s}), \quad \forall\ m \in \{1,\dots,M\},\ k \in \mathcal{S}^m, \tag{A.6d}$$

$$\sum_{k\in\mathcal{S}^m} z^m_{ka}(t) = \sum_{k\in\mathcal{S}^{m+1}} z^{m+1}_{ka}(t),$$

$$\forall\ m \in \{1,\dots,M-1\},\ t \in \{1,\dots,T+1\}, \tag{A.6e}$$

$$z^m_{ka}(t) \geq 0, \quad \forall\ m \in \{1,\dots,M\},\ k \in \mathcal{S}^m,\ a \in \mathcal{A}. \tag{A.6f}$$

This problem is identical to the finite fluid problem (2.3) (the z variables in problem (A.6) have the same meaning as the x variables in problem (2.3)); the difference is that constraint (A.6e) in problem (A.6) is expressed using additional variables (the $A_a(t)$ variables) and through a different constraint (constraint (2.3d)). Therefore, it holds that $Z^*_T(\mathbf{s}) = Z^*_{ALR(T)}(\mathbf{s})$.

To show that $Z^*_T(\mathbf{s}) = Z^*_{ALO(T)}(\mathbf{s})$, let us take the dual of the finite fluid problem (2.3):

$$\underset{\mathbf{V},\boldsymbol{\phi}}{\text{minimize}} \quad \sum_{m=1}^{M}\sum_{k\in\mathcal{S}^m} \alpha^m_k(\mathbf{s}) V^m_k(1) \tag{A.7a}$$

$$\text{subject to} \quad J^m_k(t) \geq g^m_{ka} + \phi^m_a(t) + \beta \sum_{j\in\mathcal{S}^m} p^m_{kja} J^m_j(t+1),$$

$$\forall\ m \in \{1,\dots,M\},\ k \in \mathcal{S}^m,\ t \in \{1,\dots,T\}, \tag{A.7b}$$

$$J^m_k(T+1) \geq g^m_{ka} + \phi^m_a(T+1) + \beta \sum_{j\in\mathcal{S}^m} p^m_{kja} J^m_j(T+1),$$

$$\forall\ m \in \{1,\dots,M\},\ k \in \mathcal{S}^m, \tag{A.7c}$$

$$\sum_{m=1}^{M} \phi^m_a(t) \geq 0, \quad \forall\ m \in \{1,\dots,M\},\ t \in \{1,\dots,T+1\}. \tag{A.7d}$$

To show that $Z^*_T(\mathbf{s}) \geq Z^*_{ALO(T)}(\mathbf{s})$, let $(\mathbf{V}, \boldsymbol{\phi})$ be an optimal solution of the finite

fluid dual (A.7), and define a solution $\mathbf{J}$ for ALO(T) problem (2.14) as

$$J_k^m(t) = V_k^m(t)$$

for each $m \in \{1, \ldots, M\}$, $k \in \mathcal{S}^m$ and $t \in \{1, \ldots, T+1\}$. Now observe that for any $\mathbf{s} \in \mathcal{S}$ and $t \in \{1, \ldots, T\}$, we have

$$\begin{aligned}
\sum_{m=1}^{M} J_{s^m}^m(t) &= \sum_{m=1}^{M} V_k^m(t) \\
&\geq \sum_{m=1}^{M} \left(g_{s^m a}^m + \phi_a^m(t) + \beta \sum_{j \in \mathcal{S}^m} p_{s^m ja}^m J_j^m(t+1) \right) \\
&= \sum_{m=1}^{M} g_{s^m a}^m + \sum_{m=1}^{M} \phi_a^m(t) + \beta \sum_{m=1}^{M} \sum_{j \in \mathcal{S}^m} p_{s^m ja}^m J_j^m(t+1) \\
&\geq \sum_{m=1}^{M} g_{s^m a}^m + \beta \sum_{m=1}^{M} \sum_{j \in \mathcal{S}^m} p_{s^m ja}^m J_j^m(t+1)
\end{aligned}$$

where the first step follows by our construction of $\mathbf{J}$; the second step, through constraint (A.7b) that $(\mathbf{V}, \boldsymbol{\phi})$ satisfies; the third step by distributing the sum; and the last step by constraint (A.7d). A similar argument can be used to establish that $\mathbf{J}$ satisfies

$$\sum_{m=1}^{M} J_{s^m}^m(T+1) \geq \sum_{m=1}^{M} g_{s^m a}^m + \beta \sum_{m=1}^{M} \sum_{j \in \mathcal{S}^m} p_{s^m ja}^m J_j^m(T+1).$$

The solution $\mathbf{J}$ is therefore feasible for ALO(T) problem (2.14); moreover, its objective value is

$$\begin{aligned}
\sum_{m=1}^{M} \sum_{k \in \mathcal{S}^m} \alpha_k^m(\mathbf{s}) J_k^m(1) &= \sum_{m=1}^{M} \sum_{k \in \mathcal{S}^m} \alpha_k^m(\mathbf{s}) V_k^m(1) \\
&= Z_T^*(\mathbf{s}).
\end{aligned}$$

We therefore must have that $Z_T^*(\mathbf{s}) \geq Z_{ALO(T)}^*(\mathbf{s})$.

To show that $Z_T^*(\mathbf{s}) \leq Z_{ALO(T)}^*(\mathbf{s})$, let $\mathbf{J}$ be an optimal solution of the ALO(T)

problem (2.14). Define a solution for the finite fluid dual (A.7) as follows:

$$
\begin{aligned}
&V_k^m(t) = J_k^m(t), \quad \forall\ m \in \{1,\ldots,M\},\ k \in \mathcal{S}^m,\\
&\phi_a^m(t) = \min_{k\in\mathcal{S}^m}\left(J_k^m(T+1) - g_{ka}^m - \beta\sum_{j\in\mathcal{S}^m} p_{kja}^m J_j^m(t+1)\right),\\
&\qquad \forall m \in \{1,\ldots,M\},\ a \in \mathcal{A},\ t \in \{1,\ldots,T\},\\
&\phi_a^m(T+1) = \min_{k\in\mathcal{S}^m}\left(J_k^m(T+1) - g_{ka}^m - \beta\sum_{j\in\mathcal{S}^m} p_{kja}^m J_j^m(T+1)\right),\\
&\qquad \forall m \in \{1,\ldots,M\},\ a \in \mathcal{A}.
\end{aligned}
$$

Observe that by definition of $\phi_a^m(t)$ for $t < T+1$, we have that

$$\phi_a^m(t) \le J_k^m(t) - g_{ka}^m - \beta\sum_{j\in\mathcal{S}^m} p_{kja}^m J_j^m(t+1),$$

for any $m \in \{1,\ldots,M\}$ and $k \in \mathcal{S}^m$, which we can re-arrange to obtain

$$J_k^m(t) \ge g_{ka}^m + \phi_a^m(t) + \beta\sum_{j\in\mathcal{S}^m} p_{kja}^m J_j^m(t+1),$$

or equivalently

$$V_k^m(t) \ge g_{ka}^m + \phi_a^m(t) + \beta\sum_{j\in\mathcal{S}^m} p_{kja}^m V_j^m(t+1).$$

Similarly, for $T+1$, we have

$$\phi_a^m(T+1) \le J_k^m(T+1) - g_{ka}^m - \beta\sum_{j\in\mathcal{S}^m} p_{kja}^m J_j^m(T+1),$$

which can be re-arranged to get

$$J_k^m(T+1) \ge g_{ka}^m + \phi_a^m(T+1) + \beta\sum_{j\in\mathcal{S}^m} p_{kja}^m J_j^m(T+1),$$

or equivalently

$$V_k^m(T+1) \geq g_{ka}^m + \phi_a^m(T+1) + \beta \sum_{j \in \mathcal{S}^m} p_{kja}^m V_j^m(T+1);$$

this verifies constraints (A.7b) and (A.7c).

To verify the last constraint (A.7d), we have for any given $a \in \mathcal{A}$ and $t \in \{1, \ldots, T\}$ that

$$\begin{aligned}\sum_{m=1}^{M} \phi_a^m(t) &= \sum_{m=1}^{M} \min_{k \in \mathcal{S}^m} \left\{ J_k^m(t) - g_{ka}^m - \beta \sum_{j \in \mathcal{S}^m} p_{kja}^m J_j^m(t+1) \right\} \\ &= \min_{\bar{\mathbf{s}} \in \mathcal{S}} \left\{ \sum_{m=1}^{M} \left(J_{\bar{s}^m}^m(t) - g_{\bar{s}^m a}^m - \beta \sum_{j \in \mathcal{S}^m} p_{\bar{s}^m ja}^m J_j^m \right) \right\},\end{aligned}$$

where, as in the proof of part (a) of Theorem 2 (see Section A.1.6), the order of summation and minimization can be interchanged because the minimizations are independent of each other. Since $\mathbf{J}$ is feasible for problem (2.14), it must be that

$$\sum_{m=1}^{M} J_{\bar{s}^m}^m(t) - \sum_{m=1}^{M} g_{\bar{s}^m a}^m - \beta \sum_{m=1}^{M} \sum_{j \in \mathcal{S}^m} p_{\bar{s}^m ja}^m J_j^m(t+1) \geq 0,$$

for all $\mathbf{s} \in \mathcal{S}$; it therefore follows that

$$\min_{\bar{\mathbf{s}} \in \mathcal{S}} \left\{ \sum_{m=1}^{M} J_{\bar{s}^m}^m(t) - \sum_{m=1}^{M} g_{\bar{s}^m a}^m - \beta \sum_{m=1}^{M} \sum_{j \in \mathcal{S}^m} p_{\bar{s}^m ja}^m J_j^m(t+1) \right\} \geq 0,$$

which establishes that

$$\sum_{m=1}^{M} \phi_a^m(t) \geq 0.$$

Similar steps can be used to establish that

$$\sum_{m=1}^{M} \phi_a^m(T+1) \geq 0$$

for $a \in \mathcal{A}$. Thus, $(\mathbf{V}, \boldsymbol{\phi})$ is a feasible solution for the finite fluid dual (A.7); it is also easy to see that its objective value is exactly $Z_{ALO(T)}^*(\mathbf{s})$. This establishes that

$Z_T^*(\mathbf{s}) \leq Z_{ALO(T)}^*(\mathbf{s})$.

Together with $Z_T^*(\mathbf{s}) \geq Z_{ALO(T)}^*(\mathbf{s})$, this establishes that $Z_T^*(\mathbf{s}) = Z_{ALO(T)}^*(\mathbf{s})$. This concludes the proof. □

A.1.9 Proof of Proposition 9

The proof follows by showing that $Z_{BNM}^*(\mathbf{s}) \leq Z_{ALR}^*(\mathbf{s})$ and $Z_{BNM}^*(\mathbf{s}) \geq Z_{ALR}^*(\mathbf{s})$. To do this, we follow our approach in the other proofs of using one problem's optimal solution to construct a feasible solution for the other problem that has the same objective value as the first problem's optimal solution. The most difficult part of the proof is deriving a suitable feasible solution for one problem from the optimal solution of the other; verifying that the solutions are feasible is straightforward, but somewhat laborious. In what follows, we will present several identities that are useful in verifying feasibility and then provide the correct feasible solutions, but in the interest of space, we will omit the verification of the solutions.

Let us derive a few properties of problems (2.12) and (2.15) that will be useful to us. First of all, notice that for any $\mathbf{z}$ feasible for problem (2.12), for a fixed $m \in \{1, \ldots, M\}$, by summing constraint (2.12b) over all states $j \in \mathcal{S}^m$, we have

$$\sum_{j \in \mathcal{S}^m} \sum_{a'=1}^{M} z_{ja'}^m - \beta \sum_{j \in \mathcal{S}^m} \sum_{k \in \mathcal{S}^m} \sum_{a=1}^{M} p_{kja}^m z_{ka}^m = \sum_{j \in \mathcal{S}^m} \alpha_j^m(\mathbf{s}),$$

which is equal to

$$\sum_{j \in \mathcal{S}^m} \sum_{a'=1}^{M} z_{ja'}^m - \beta \sum_{k \in \mathcal{S}^m} \sum_{a=1}^{M} \left(\sum_{j \in \mathcal{S}^m} p_{kja}^m \right) \cdot z_{ka}^m = \sum_{j \in \mathcal{S}^m} \alpha_j^m(\mathbf{s}).$$

Since $\sum_{j \in \mathcal{S}^m} p_{kja}^m = 1$ and $\sum_{j \in \mathcal{S}^m} \alpha_j^m(\mathbf{s}) = 1$, this reduces to

$$\sum_{j \in \mathcal{S}^m} \sum_{a'=1}^{M} z_{ja'}^m - \beta \sum_{k \in \mathcal{S}^m} \sum_{a=1}^{M} z_{ka}^m = 1.$$

Observe that the two sums are identical; by dividing through by $(1 - \beta)$, we obtain

the following identity:

$$\sum_{k\in\mathcal{S}^m}\sum_{a=1}^{M} z_{ka}^m = \frac{1}{1-\beta}, \qquad \forall\ m\in\{1,\dots,M\}. \tag{A.8}$$

Observe that by following the same steps, we can derive a similar identity for the $\mathbf{w}$ variables of problem (2.15) using constraint (2.15c):

$$\sum_{k\in\mathcal{S}^m}\sum_{a'\in\{0,1\}} w_{ka'}^m = \frac{1}{1-\beta}, \qquad \forall\ m\in\{1,\dots,M\}. \tag{A.9}$$

Lastly, notice that by combining constraint (2.15b) and identity (A.9) for a fixed $m\in\{1,\dots,M\}$, we get that

$$\sum_{m'=1}^{M}\sum_{k\in\mathcal{S}^{m'}} w_{k1}^{m'} = \sum_{k\in\mathcal{S}^m}\sum_{a\in\{0,1\}} w_{ka}^m.$$

Notice that both sums contain the sum $\sum_{k\in\mathcal{S}^m} w_{k1}^m$; by subtracting it out, we obtain the following identity:

$$\sum_{m'\neq m}\sum_{k\in\mathcal{S}^{m'}} w_{k1}^{m'} = \sum_{k\in\mathcal{S}^m} w_{k0}^m, \qquad \forall\ m\in\{1,\dots,M\}. \tag{A.10}$$

Having defined these identities, we are ready to move on with the proof.

To prove that $Z_{ALR}^*(\mathbf{s}) \geq Z_{BNM}^*(\mathbf{s})$, suppose that $\mathbf{w}$ is an optimal solution to problem (2.15). We will proceed in two different cases.

Case 1 of $Z_{ALR}^*(\mathbf{s}) \geq Z_{BNM}^*(\mathbf{s})$. Suppose that $\sum_{j\in\mathcal{S}^m} w_{j0}^m > 0$ for every $m\in\{1,\dots,M\}$. Define a solution to problem (2.12) as

$$\begin{aligned} z_{km}^m &= w_{k1}^m, && \forall\ m\in\{1,\dots,M\},\ k\in\mathcal{S}^m,\\ z_{ka}^m &= \frac{w_{k0}^m}{\sum_{j\in\mathcal{S}^m} w_{j0}^m}\cdot\sum_{j\in\mathcal{S}^a} w_{j1}^a && \forall\ m\in\{1,\dots,M\},\ a\neq m,\ k\in\mathcal{S}^m. \end{aligned}$$

It can be shown that $\mathbf{z}$ is feasible for problem (2.12) and that its objective in prob-

lem (2.12) is exactly $Z^*_{BNM}(\mathbf{s})$.

Case 2 of $Z^*_{ALR}(\mathbf{s}) \geq Z^*_{BNM}(\mathbf{s})$. In the second case, we suppose that there exists an $\tilde{m}$ such that $\sum_{j \in \mathcal{S}^m} w^{\tilde{m}}_{j0} = 0$. In this case, the expected discounted frequency with which we do not active bandit $\tilde{m}$ when it is in any of its states is zero; intuitively, this means that we are always activating bandit $\tilde{m}$.

Armed with this intuition and the above identity, let us define a solution $\mathbf{z}$ for problem (2.12) as

$$\begin{aligned}
&z^m_{ka} = 0, && \forall\ m \in \{1, \ldots, M\},\ k \in \mathcal{S}^m,\ a \neq \tilde{m},\\
&z^m_{k\tilde{m}} = w^m_{k0}, && \forall\ m \neq \tilde{m},\ k \in \mathcal{S}^{\tilde{m}},\\
&z^{\tilde{m}}_{k\tilde{m}} = w^{\tilde{m}}_{k1}, && \forall k \in \mathcal{S}^{\tilde{m}}.
\end{aligned}$$

As in Case 1, it can then be shown that $\mathbf{z}$ is feasible for problem (2.12) and that its objective in problem (2.12) is exactly $Z^*_{BNM}(\mathbf{s})$.

This proves that $Z^*_{BNM}(\mathbf{s}) \leq Z^*_{ALR}(\mathbf{s})$; we now turn our attention to $Z^*_{BNM}(\mathbf{s}) \geq Z^*_{ALR}(\mathbf{s})$. To prove this inequality, suppose that we have a solution $\mathbf{z}$ to problem (2.12). Define the solution $\mathbf{w}$ to problem (2.15) as

$$\begin{aligned}
&w^m_{k0} = \sum_{a \neq m} z^m_{ka}, && \forall\ m \in \{1, \ldots, M\},\ k \in \mathcal{S}^m,\\
&w^m_{k1} = z^m_{km}, && \forall\ m \in \{1, \ldots, M\},\ k \in \mathcal{S}^m.
\end{aligned}$$

It can be shown that $\mathbf{w}$ is feasible for problem (2.15) and that its objective in problem (2.15) is exactly $Z^*_{ALR}(\mathbf{s})$.

From here, since $Z^*_{ALR}(\mathbf{s}) \leq Z^*_{BNM}(\mathbf{s})$ and $Z^*_{ALR}(\mathbf{s}) \geq Z^*_{BNM}(\mathbf{s})$, it follows that $Z^*_{ALR}(\mathbf{s}) = Z^*_{BNM}(\mathbf{s})$, as required. $\square$

A.2 Counterexample to show that $Z^*(\mathbf{s}) \leq J^*(\mathbf{s})$ does not always hold

To develop the counterexample, we proceed in two steps. First, we develop a useful bound that relates the optimal value of a truncated fluid formulation and the infinite fluid formulation (problem (2.2)). Then, we describe a specific problem instance where the numerical values of the optimal DP value function and the optimal value of the finite fluid formulation, together with the bound we just developed, allow us to conclude that $Z^*(\mathbf{s}) \leq J^*(\mathbf{s})$ does not hold.

A.2.1 Bound

Clearly, we are not able to solve the optimization problem (2.2), since it is a problem with a countably infinite number of constraints and variables. However, if we only include the decision variables and constraints corresponding to $t = 1$ to $t = T$, and truncate the objective function to the finite sum $\sum_{t=1}^{T}\sum_{m=1}^{M}\sum_{k=1}^{n}\sum_{a=1}^{M}\beta^{t-1}g_{ka}^{m}x_{ka}^{m}(t)$, we obtain a finite formulation that we are able to solve. (This is in fact problem (2.3) without the $T+1$ objective term that accounts for the system's evolution from $t = T+1$ on.) Let $Z^*_{T,\text{trunc}}(\mathbf{s})$ be the optimal value of this truncated problem. The finite-length optimal solution $(\mathbf{x}, \mathbf{A})$ that attains this objective value can be easily extended for times $t > T$ (e.g., by setting $A_1(t) = 1$ for $t > T$, $x_{ka}^{m}(t)$ and $A_a(t)$ are uniquely determined for $t > T$) to yield a completed, infinite-length solution. Using this infinite-length solution, we see that

$$Z^*_{T,\text{trunc}}(\mathbf{s}) = \sum_{t=1}^{T}\sum_{m=1}^{M}\sum_{k=1}^{n}\sum_{a=1}^{M}\beta^{t-1}g_{ka}^{m}x_{ka}^{m}(t) \leq \sum_{t=1}^{\infty}\sum_{m=1}^{M}\sum_{k=1}^{n}\sum_{a=1}^{M}\beta^{t-1}g_{ka}^{m}x_{ka}^{m}(t) \leq Z^*(\mathbf{s}),$$

where the first step follows by the definition of $Z^*_T(\mathbf{s})$ as the optimal value of the truncated formulation, the second by the fact that the value of the sum can only increase when it is extended from a finite one to an infinite one (all g_{ka}^{m} and $x_{ka}^{m}(t)$ values are nonnegative), and the third by the fact that the completed, infinite-length

solution is a feasible solution to the infinite fluid formulation (2.2).

A.2.2 Instance

Consider a decomposable MDP with $M = 3$, $\mathcal{S}^1 = \mathcal{S}^2 = \mathcal{S}^3 = \{1, 2, 3\}$, with the following reward data:

$$\mathbf{g}_1^1 = \begin{bmatrix} 3.3323 \\ 0.2765 \\ 2.2476 \end{bmatrix}, \ \mathbf{g}_2^2 = \begin{bmatrix} 2.6230 \\ 8.9540 \\ 4.5576 \end{bmatrix}, \mathbf{g}_3^3 = \begin{bmatrix} 2.8975 \\ 7.8568 \\ 0.8381 \end{bmatrix}, \tag{A.11}$$

$$\mathbf{g}_a^m = \mathbf{0}, \ \forall m \in \{1, 2, 3\}, \ a \in \{1, 2, 3\}, \ a \neq m, \tag{A.12}$$

and the following probability transition data:

$$\mathbf{p}_1^1 = \begin{bmatrix} 0.0540 & 0.3790 & 0.5670 \\ 0.0190 & 0.0390 & 0.9420 \\ 0.3120 & 0.5330 & 0.1550 \end{bmatrix}, \tag{A.13}$$

$$\mathbf{p}_1^2 = \begin{bmatrix} 0.8560 & 0 & 0.1440 \\ 0.1590 & 0.3150 & 0.5260 \\ 0.0320 & 0.8050 & 0.1630 \end{bmatrix}, \tag{A.14}$$

$$\mathbf{p}_1^3 = \begin{bmatrix} 0.0740 & 0.9020 & 0.0240 \\ 0.3440 & 0.0200 & 0.6360 \\ 0.5820 & 0.3750 & 0.0430 \end{bmatrix} \tag{A.15}$$

$$\mathbf{p}_a^m = \mathbf{I}, \ \forall m \in \{1, 2, 3\}, \ a \in \{1, 2, 3\}, \ a \neq m. \tag{A.16}$$

As we will see in Section 2.5, this problem actually corresponds to a regular multi-armed bandit problem with three arms.

Suppose that the discount factor is 0.9 and the initial state $\mathbf{s}$ is set to $(3, 3, 1)$. Solving the truncated version of problem (2.2) with $T = 100$ for initial state $\mathbf{s}$, we obtain an objective value of $Z^*_{T,\text{trunc}}(\mathbf{s}) = 57.7134$, while the optimal DP value function

is $J^*(\mathbf{s}) = 57.3812$. By the bound above, we have

$$J^*(\mathbf{s}) = 57.3812 < 57.7134 = Z^*_{T,\text{trunc}}(\mathbf{s}) \leq Z^*(\mathbf{s}),$$

i.e., that $J^*(\mathbf{s}) < Z^*(\mathbf{s})$. This allows us to conclude that $J^*(\mathbf{s}) \geq Z^*(\mathbf{s})$ does not always hold.

A.3 Derivation of alternate Lagrangian relaxation

In this section, we derive the ALR formulation. The steps that we follow here are essentially the same as those used in [2] to derive the Lagrangian relaxation formulation, applied to the specific weakly-coupled MDP that is at the heart of the ALR. For completeness, we show the main steps of the derivation here.

The optimal value function for the true MDP of interest satisfies the following Bellman equation:

$$J^*(\mathbf{s}) = \max_{\substack{(a^1,\ldots,a^M)\in\mathcal{A}\times\cdots\times\mathcal{A}:\\ \mathbb{I}\{a^m=a\}-\mathbb{I}\{a^{m+1}=a\}=0,\\ \forall m\in\{1,\ldots,M-1\},\ a\in\mathcal{A}}} \left(\sum_{m=1}^{M} g^m_{s^m a^m} + \beta \cdot \sum_{\bar{\mathbf{s}}\in\mathcal{S}} \left(\prod_{m=1}^{M} p^m_{s^m \bar{s}^m a^m} \right) J^*(\bar{\mathbf{s}}) \right). \quad \text{(A.17)}$$

In the Lagrangian relaxation approach [58, 2], we dualize the action consistency constraint on the action vectors $(a^1, \ldots, a^M)$. For each constraint in the maximization, we introduce a Lagrange multiplier $\lambda^m_a \in \mathbb{R}$, and penalize the violation of the corresponding (m, a) constraint in the Bellman iteration. We obtain a new value function

$J^{\boldsymbol{\lambda}}(\mathbf{s})$ which satisfies the following modified Bellman equation:

$$J^{\boldsymbol{\lambda}}(\mathbf{s}) = \max_{(a^1,\ldots,a^M)\in\mathcal{A}\times\cdots\times\mathcal{A}} \left(\sum_{m=1}^{M} g^m_{s^m a^m} + \beta \cdot \sum_{\bar{\mathbf{s}}\in\mathcal{S}} \left(\prod_{m=1}^{M} p^m_{s^m \bar{s}^m a^m} \right) J^{\boldsymbol{\lambda}}(\bar{\mathbf{s}}) \right.$$
$$\left. - \sum_{m=1}^{M-1} \sum_{a\in\mathcal{A}} \lambda^m_a (\mathbb{I}\{a^m = a\} - \mathbb{I}\{a^{m+1} = a\}) \right) \tag{A.18}$$
$$= \max_{(a^1,\ldots,a^M)\in\mathcal{A}\times\cdots\times\mathcal{A}} \left(\sum_{m=1}^{M} g^m_{s^m a^m} + \beta \cdot \sum_{\bar{\mathbf{s}}\in\mathcal{S}} \left(\prod_{m=1}^{M} p^m_{s^m \bar{s}^m a^m} \right) J^{\boldsymbol{\lambda}}(\bar{\mathbf{s}}) \right.$$
$$\left. - \sum_{m=1}^{M-1} (\lambda^m_{a^m} - \lambda^m_{a^{m+1}}) \right) \tag{A.19}$$

We will show that the solution to the above Bellman equation is of the form

$$J^{\boldsymbol{\lambda}}(\mathbf{s}) = \sum_{m=1}^{M} V^{m,\boldsymbol{\lambda}}(s^m), \tag{A.20}$$

where each $V^{m,\boldsymbol{\lambda}}$ is a component-wise value function satisfying

$$V^{m,\boldsymbol{\lambda}}(s^m) = \max_{a\in\mathcal{A}} \left(g^m_{s^m a} + \beta \sum_{\bar{s}^m\in\mathcal{S}^m} p^m_{s^m \bar{s}^m a^m} V^{m,\boldsymbol{\lambda}}(\bar{s}^m) - \mathbb{I}\{m < M\} \cdot \lambda^m_a + \mathbb{I}\{m > 1\} \cdot \lambda^{m-1}_a \right) \tag{A.21}$$

To see this, we will show that the above form satisfies equation (A.20). We have

$$\max_{(a^1,\ldots,a^M)\in\mathcal{A}\times\cdots\times\mathcal{A}} \left(\sum_{m=1}^{M} g^m_{s^m a^m} + \beta \cdot \sum_{\bar{\mathbf{s}}\in\mathcal{S}} \left(\prod_{m=1}^{M} p^m_{s^m \bar{s}^m a^m} \right) \left(\sum_{m=1}^{M} V^{m,\boldsymbol{\lambda}}(\bar{s}^m) \right) - \sum_{m=1}^{M-1} (\lambda^m_{a^m} - \lambda^m_{a^{m+1}}) \right)$$
$$= \max_{(a^1,\ldots,a^M)\in\mathcal{A}\times\cdots\times\mathcal{A}} \left(\sum_{m=1}^{M} g^m_{s^m a^m} + \beta \cdot \sum_{m=1}^{M} \sum_{\bar{s}^m\in\mathcal{S}^m} p^m_{s^m \bar{s}^m a^m} V^{m,\boldsymbol{\lambda}}(\bar{s}^m) - \sum_{m=1}^{M-1} (\lambda^m_{a^m} - \lambda^m_{a^{m+1}}) \right)$$
$$= \sum_{m=1}^{M} \max_{a^m\in\mathcal{A}} \left(g^m_{s^m a^m} + \beta \sum_{\bar{s}^m\in\mathcal{S}^m} p^m_{s^m \bar{s}^m a^m} V^{m,\boldsymbol{\lambda}}(\bar{s}^m) - \mathbb{I}\{m < M\} \cdot \lambda^m_a + \mathbb{I}\{m > 1\} \cdot \lambda^{m-1}_a \right)$$
$$= \sum_{m=1}^{M} V^{m,\boldsymbol{\lambda}}(s^m),$$

where the first equality follows by the linearity of expectation (the expression

$$\sum_{\bar{\mathbf{s}} \in \mathcal{S}} \left(\prod_{m=1}^{M} p^m_{s^m \bar{s}^m a^m} \right) \left(\sum_{m=1}^{M} V^{m,\boldsymbol{\lambda}}(\bar{s}^m) \right)$$

can be viewed as the expectation of a sum of random variables that correspond to each component's value function at a random next state); the second by the fact that the maximizations over each of the a^m variables are independent of each other; and the third by definition of the $V_k^{m,\boldsymbol{\lambda}}$'s.

To now derive the ALR formulation, let $\mathbf{s}$ be the initial state. The value of $\mathbf{s}$ is $J^{\boldsymbol{\lambda}}(\mathbf{s}) = \sum_{m=1}^{M} V^{m,\boldsymbol{\lambda}}(s^m)$. Each component value function $V^{m,\boldsymbol{\lambda}}$, described by the Bellman equation in equation (A.21), can be evaluated at s^m by solving the following linear optimization problem, where $\boldsymbol{\lambda}$ is fixed (i.e., not a decision variable):

$$\begin{aligned}
V^{m,\boldsymbol{\lambda}}(s^m) \;= \underset{\mathbf{V}^m}{\text{minimize}} \quad & \sum_{k \in \mathcal{S}^m} \alpha^m_k(\mathbf{s}) V^m_k && \text{(A.22a)} \\
\text{subject to} \quad & V^m_k \geq g^m_{ka} - \mathbb{I}\{m < M\} \cdot \lambda^m_a + \mathbb{I}\{m > 1\} \cdot \lambda^{m-1}_a \\
& \qquad + \beta \cdot \sum_{j \in \mathcal{S}^m} p^m_{kja} V^m_j, && \text{(A.22b)} \\
& \forall\ k \in \mathcal{S}^m,\ a \in \mathcal{A}. && \text{(A.22c)}
\end{aligned}$$

The value $J^{\boldsymbol{\lambda}}(\mathbf{s})$ can then be expressed as the optimal value of the following optimization problem, which combines the above component-wise optimization problems into one problem:

$$\begin{aligned}
J^{\boldsymbol{\lambda}}(\mathbf{s}) \;= \underset{\mathbf{V}}{\text{minimize}} \quad & \sum_{m=1}^{M} \sum_{k \in \mathcal{S}^m} \alpha^m_k(\mathbf{s}) V^m_k && \text{(A.23a)} \\
\text{subject to} \quad & V^m_k \geq g^m_{ka} - \mathbb{I}\{m < M\} \cdot \lambda^m_a + \mathbb{I}\{m > 1\} \cdot \lambda^{m-1}_a \\
& \qquad + \beta \cdot \sum_{j \in \mathcal{S}^m} p^m_{kja} V^m_j, && \text{(A.23b)} \\
& \forall\ m \in \{1, \ldots, M\},\ k \in \mathcal{S}^m,\ a \in \mathcal{A}. && \text{(A.23c)}
\end{aligned}$$

As in [2], it can be shown that the optimal value of the above problem, which is equal

to $J^{\boldsymbol{\lambda}}(\mathbf{s})$, is an upper bound on $J^*(\mathbf{s})$. We now seek to find the tightest such upper bound, that is, $\min_{\boldsymbol{\lambda}} J^{\boldsymbol{\lambda}}(\mathbf{s})$. This can be accomplished by optimizing over $\boldsymbol{\lambda}$ as an additional decision variable in the above optimization problem:

$$\underset{\mathbf{V},\boldsymbol{\lambda}}{\text{minimize}} \quad \sum_{m=1}^{M} \sum_{k \in \mathcal{S}^m} \alpha_k^m(\mathbf{s}) V_k^m \tag{A.24a}$$

$$\text{subject to} \quad V_k^m \geq g_{ka}^m - \mathbb{I}\{m < M\} \cdot \lambda_a^m + \mathbb{I}\{m > 1\} \cdot \lambda_a^{m-1} + \beta \cdot \sum_{j \in \mathcal{S}^m} p_{kja}^m V_j^m,$$

$$\forall\ m \in \{1, \ldots, M\},\ k \in \mathcal{S}^m,\ a \in \mathcal{A}. \tag{A.24b}$$

The above formulation is exactly the ALR formulation (2.11).

Bibliography

[1] D. Adelman. Dynamic bid prices in revenue management. *Operations Research*, 55(4):647–661, 2007.

[2] D. Adelman and A. J. Mersereau. Relaxations of weakly coupled stochastic dynamic programs. *Operations Research*, 56(3):712–727, 2008.

[3] H. Akaike. A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, 19(6):716–723, 1974.

[4] G. M. Allenby and P. E. Rossi. Marketing models of consumer heterogeneity. *Journal of Econometrics*, 89(1):57–78, 1998.

[5] E. J. Anderson and P. Nash. *Linear Programming in Infinite-Dimensional Spaces*. John Wiley & Sons, Chichester, UK, 1987.

[6] O. Baron, J. Milner, and H. Naseraldin. Facility location: A robust optimization approach. *Production and Operations Management*, 20(5):772–785, 2011.

[7] R. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, NJ, USA, 1957.

[8] R. Bellman. *Adaptive control processes: a guided tour*, volume 4. Princeton university press Princeton, 1961.

[9] A. Belloni, R. Freund, M. Selove, and D. Simester. Optimizing product line designs: Efficient methods and comparisons. *Management Science*, 54(9):1544–1552, 2008.

[10] M. E. Ben-Akiva and S. R. Lerman. *Discrete choice analysis: theory and application to travel demand*, volume 9. MIT press, 1985.

[11] F. Bernstein, A. G. Kök, and L. Xie. Dynamic assortment customization with limited inventories. *Manufacturing & Service Operations Management*, 17(4):538–553, 2015.

[12] D. P. Bertsekas. *Dynamic programming and optimal control*, volume 1. Athena Scientific, 1995.

[13] D. P. Bertsekas and J. N. Tsitsiklis. *Neuro-Dynamic Programming*. Athena Scientific, 1996.

[14] D. Bertsimas. The achievable region method in the optimal control of queueing systems; formulations, bounds and policies. *Queueing Systems: Theory and Applications*, 21(3-4):337–389, 1995.

[15] D. Bertsimas, D. B. Brown, and C. Caramanis. Theory and applications of robust optimization. *SIAM Review*, 53(3):464–501, 2011.

[16] D. Bertsimas and N. Kallus. From predictive to prescriptive analytics. *arXiv preprint arXiv:1402.5481*, 2015.

[17] D. Bertsimas, E. Litvinov, X. A. Sun, J. Zhao, and T. Zheng. Adaptive robust optimization for the security constrained unit commitment problem. *IEEE Transactions on Power Systems*, 28(1):52–63, 2013.

[18] D. Bertsimas and J. Niño-Mora. Conservation laws, extended polymatroids and multiarmed bandit problems; a polyhedral approach to indexable systems. *Mathematics of Operations Research*, 21(2):257–306, 1996.

[19] D. Bertsimas and J. Niño-Mora. Restless bandits, linear programming relaxations, and a primal-dual index heuristic. *Operations Research*, 48(1):80–90, 2000.

[20] D. Bertsimas, J. Silberholz, and T. Trikalinos. Decision-making under competing interpretations of the evidence: Application in prostate cancer screening. *Submitted for publication*, 2015.

[21] D. Bertsimas and M. Sim. The price of robustness. *Operations Research*, 52(1):35–53, 2004.

[22] D. Bertsimas and A. Thiele. A robust optimization approach to inventory theory. *Operations Research*, 54(1):150–168, 2006.

[23] D. Bertsimas and J. N. Tsitsiklis. *Introduction to linear optimization*, volume 6. Athena Scientific, Belmont, MA, 1997.

[24] J. Bezanson, S. Karpinski, V. B. Shah, and A. Edelman. Julia: A fast dynamic language for technical computing. *arXiv preprint arXiv:1209.5145*, 2012.

[25] J. Blanchet, G. Gallego, and V. Goyal. A Markov chain approximation to choice modeling. *Submitted*, 2013. Available at `http://www.columbia.edu/~vg2277/MC_paper.pdf`.

[26] T. Bortfeld, T. C. Y. Chan, A. Trofimov, and J. N. Tsitsiklis. Robust management of motion uncertainty in intensity-modulated radiation therapy. *Operations Research*, 56(6):1461–1473, 2008.

[27] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends®*.

[28] H. Bozdogan. Model selection and akaike's information criterion (aic): The general theory and its analytical extensions. *Psychometrika*, 52(3):345–370, 1987.

[29] L. Breiman, J. Friedman, C. J. Stone, and R. A. Olshen. *Classification and regression trees*. CRC press, 1984.

[30] J. J. M. Bront, I. Méndez-Díaz, and G. Vulcano. A column generation algorithm for choice-based network revenue management. *Operations Research*, 57(3):769–784, 2009.

[31] Celect, Inc., 2014. Accessed February 11, 2015; available at `http://www.celect.net`.

[32] T. C. Y. Chan, Z.-J. M. Shen, and A. Siddiq. Robust facility location under demand location uncertainty. *arXiv preprint arXiv:1507.04397*, 2015.

[33] K. D. Chen and W. H. Hausman. Technical note: Mathematical properties of the optimal product line selection problem using choice-based conjoint analysis. *Management Science*, 46(2):327–332, 2000.

[34] X. Chen, Z. Owen, C. Pixton, and D. Simchi-Levi. A statistical learning approach to personalization in revenue management. *Available at SSRN 2579462*, 2015.

[35] E. G. Coffman and I. Mitrani. A characterization of waiting time performance realizable by single-server queues. *Operations Research*, 28(3):810–821, 1980.

[36] J. Davis, G. Gallego, and H. Topaloglu. Assortment planning under the multinomial logit model with totally unimodular constraint structures. Technical report, Department of IEOR, Columbia University. Available at `http://www.columbia.edu/~gmg2/logit_const.pdf`, 2013.

[37] J. M. Davis, G. Gallego, and H. Topaloglu. Assortment optimization under variants of the nested logit model. *Operations Research*, 62(2):250–273, 2014.

[38] D. P. de Farias and B. Van Roy. The linear programming approach to approximate dynamic programming. *Operations Research*, 51(6):850–865, 2003.

[39] D. P. de Farias and B. Van Roy. On constraint sampling in the linear programming approach to approximate dynamic programming. *Mathematics of Operations Research*, 29(3):462–478, 2004.

[40] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society Series B*, pages 1–38, 1977.

[41] A. Désir and V. Goyal. Near-optimal algorithms for capacity constrained assortment optimization. *Available at SSRN 2543309*, 2014.

[42] G. Dobson and S. Kalish. Positioning and pricing a product line. *Marketing Science*, 7(2):107–125, 1988.

[43] G. Dobson and S. Kalish. Heuristics for pricing and positioning a product-line using conjoint and cost data. *Management Science*, 39(2):160–175, 1993.

[44] V. F. Farias, S. Jagabathula, and D. Shah. A nonparametric approach to modeling choice with limited data. *Management Science*, 59(2):305–322, 2013.

[45] A. Federgruen and H. Groenevelt. Characterization and optimization of achievable performance in general queueing systems. *Operations Research*, 36(5):733–741, 1988.

[46] J. B. Feldman and H. Topaloglu. Revenue Management Under the Markov Chain Choice Model. *Working paper*, 2014. Available at `http://people.orie.cornell.edu/huseyin/publications/mc_revenue.pdf`.

[47] J. B. Feldman and H. Topaloglu. Capacity constraints across nests in assortment optimization under the nested logit model. *Operations Research*, forthcoming, 2015. Available at `http://legacy.orie.cornell.edu/huseyin/publications/nested_capacitated_full.pdf`.

[48] M. Fisher and R. Vaidyanathan. Which Products Should You Stock? A new approach to assortment planning turns an art into a science. *Harvard Business Review*, pages 108–118, 2012.

[49] G. Gallego and H. Topaloglu. Constrained assortment optimization for the nested logit model. *Management Science*, 60(10):2583–2601, 2014.

[50] A. Ghate and R. L. Smith. A linear programming approach to nonstationary infinite-horizon markov decision processes. *Operations Research*, 61(2):413–425, 2013.

[51] D. Goldfarb and G. Iyengar. Robust portfolio selection problems. *Mathematics of Operations Research*, 28(1):1–38, 2003.

[52] D. Goldfarb and S. Ma. Fast multiple-splitting algorithms for convex optimization. *SIAM Journal on Optimization*, 22(2):533–556, 2012.

[53] N. Golrezaei, H. Nazerzadeh, and P. Rusmevichientong. Real-time optimization of personalized assortments. *Management Science*, 60(6):1532–1551, 2014.

[54] P. E. Green and A. M. Krieger. Models and heuristics for product line selection. *Marketing Science*, 4(1):1–19, 1985.

[55] P. E. Green and A. M. Krieger. Conjoint analysis with product-positioning applications. In J. Eliashberg and G. L. Lilien, editors, *Handbooks in Operations Research and Management Science*, volume 5, pages 467–515. Elsevier, 1993.

[56] Gurobi Optimization, Inc. Gurobi Optimizer Reference Manual, 2015.

[57] T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning*. Springer, 2009.

[58] J. T. Hawkins. *A Lagrangian decomposition approach to weakly coupled dynamic optimization problems and its applications*. PhD thesis, Massachusetts Institute of Technology, 2003.

[59] D. P. Heyman and M. J. Sobel. *Stochastic models in operations research. Vol. 2, Stochastic optimization*. McGraw-Hill New York, 1984.

[60] R. A. Howard. *Dynamic Probabilistic Systems, Volume II: Semi-Markov and Decision Processes*. 1971.

[61] IBM. IBM – DemandTec Assortment Optimization, 2015. Accessed February 11, 2015; available at `http://www-03.ibm.com/software/products/en/assortment-optimization`.

[62] S. Jagabathula. Assortment optimization under general choice. *Available at SSRN*, 2014.

[63] S. Jasin and S. Kumar. A re-solving heuristic with bounded revenue loss for network revenue management with customer choice. *Mathematics of Operations Research*, 37(2):313–345, 2012.

[64] JDA Software Group, Inc. JDA Assortment Optimization | JDA Software, 2015. Accessed February 11, 2015; available at `http://www.jda.com/solutions/assortment-optimization/`.

[65] R. Kohli and R. Sukumar. Heuristics for product-line design using conjoint analysis. *Management Science*, 36(12):1464–1478, 1990.

[66] A. G. Kök, M. L. Fisher, and R. Vaidyanathan. Assortment planning: Review of literature and industry practice. In Narendra Agrawal and Stephen A. Smith, editors, *Retail Supply Chain Management*, volume 122 of *International Series in Operations Research & Management Science*, pages 99–153. Springer US, 2009.

[67] U. G. Kraus and C. A. Yano. Product line selection and pricing under a share-of-surplus choice model. *European Journal of Operational Research*, 150(3):653–671, 2003.

[68] I. Lee, M. A. Epelman, H. E. Romeijn, and R. L. Smith. A linear programming approach to constrained nonstationary infinite-horizon markov decision processes. Technical Report 13-01, Ann Arbor, MI: University of Michigan, Dept. of Industrial & Operations Engineering, 2013.

[69] G. Li, P. Rusmevichientong, and H. Topaloglu. The d-level nested logit model: Assortment and price optimization problems. Technical report, Cornell University, School of Operations Research and Information Engineering. Available at `http://legacy.orie.cornell.edu/~huseyin/publications/publications.html`, 2013.

[70] M. Lubin and I. Dunning. Computing in Operations Research Using Julia. *INFORMS Journal on Computing*, 27(2):238–248, 2015.

[71] R. D. McBride and F. S. Zufryden. An integer programming approach to the optimal product line selection problem. *Marketing Science*, 7(2):126–140, 1988.

[72] Oracle Corporation. Oracle Retail – World Class Commerce Solutions | Oracle, 2015. Accessed June 1, 2015; available at `https://www.oracle.com/industries/retail/index.html`.

[73] J. Osterman, W. Weaver, J. Slater, and P. Glass. Failure-proof method for intrinsic field subtraction. *Manhattan Journal of Physics*, 1(4):7–10, 1959.

[74] W. B. Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. Wiley-Interscience, 2007.

[75] M. L. Puterman. *Markov decision processes: Discrete dynamic stochastic programming*. John Wiley Chichester, 1994.

[76] H. E. Romeijn, R. L. Smith, and J. C. Bean. Duality in infinite dimensional linear programming. *Mathematical Programming*, 53(1-3):79–97, 1992.

[77] P. E. Rossi. *`bayesm`: Bayesian Inference for Marketing/Micro-econometrics*, 2012. R package version 2.2-5.

[78] P. E. Rossi and G. M. Allenby. Bayesian statistics and marketing. *Marketing Science*, 22(3):304–328, 2003.

[79] P. E. Rossi, G. M. Allenby, and R. McCulloch. *Bayesian statistics and marketing*. John Wiley & Sons, 2012.

[80] P. Rusmevichientong, Z.-J. M. Shen, and D. B. Shmoys. Dynamic assortment optimization with a multinomial logit choice model and capacity constraint. *Operations Research*, 58(6):1666–1680, 2010.

[81] P. Rusmevichientong, D. Shmoys, C. Tong, and H. Topaloglu. Assortment optimization under the multinomial logit model with random choice parameters. *Production and Operations Management*, 23(11):2023–2039, 2014.

[82] P. Rusmevichientong and H. Topaloglu. Robust assortment optimization in revenue management under the multinomial logit choice model. *Operations Research*, 60(4):865–882, 2012.

[83] D. Russo and B. Van Roy. Learning to optimize via posterior sampling. *Mathematics of Operations Research*, 39(4):1221–1243, 2014.

[84] Sawtooth Software. Advanced simulation module (asm) for product optimization v1.5. *Sawtooth Software Technical Paper Series*, 2003.

[85] C. Schön. On the optimal product line selection problem with price discrimination. *Management Science*, 56(5):896–902, 2010.

[86] C. Schön. On the product line selection problem under attraction choice models of consumer behavior. *European Journal of Operational Research*, 206(1):260–264, 2010.

[87] G. Schwarz. Estimating the dimension of a model. *Annals of Statistics*, 6(2):461–464, 1978.

[88] J. G. Shanthikumar and D. D. Yao. Multiclass queueing systems: Polymatroidal structure and optimal scheduling control. *Operations Research*, 40(3):S293–S299, 1992.

[89] K. Talluri and G. van Ryzin. Revenue management under a general discrete choice model of consumer behavior. *Management Science*, 50(1):15–33, 2004.

[90] R. H. Thaler and C. R. Sunstein. *Nudge*. Yale University Press, 2008.

[91] O. Toubia, D. I. Simester, J. R. Hauser, and E. Dahan. Fast polyhedral adaptive conjoint estimation. *Marketing Science*, 22(3):273–303, 2003.

[92] K. E. Train. Em algorithms for nonparametric estimation of mixing distributions. *Journal of Choice Modelling*, 1(1):40–69, 2008.

[93] K. E. Train. *Discrete choice methods with simulation*. Cambridge university press, 2009.

[94] B. Van Roy. Neuro-dynamic programming: Overview and recent trends. In *Handbook of Markov Decision Processes*, pages 431–459. Springer, 2002.

[95] G. van Ryzin and G. Vulcano. A market discovery algorithm to estimate a general class of nonparametric choice models. *Management Science*, 61(2):281–300, 2015.

[96] A. Wächter and L. T. Biegler. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical programming*, 106(1):25–57, 2006.

[97] A. Zeileis, T. Hothorn, and K. Hornik. Model-based recursive partitioning. *Journal of Computational and Graphical Statistics*, 17(2):492–514, 2008.

[98] F. S. Zufryden. Product line optimization by integer programming. In *Proc. Annual Meeting of ORSA/TIMS, San Diego, CA*, pages 100–114, 1982.