POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Scalable Algorithms for Discrete Choice Modeling and Assortment Optimization

CLAUDIO SOLE
Département de mathématiques et de génie industriel

Thèse présentée en vue de l'obtention du diplôme de *Philosophiæ Doctor*
Mathématiques

Octobre 2021

POLYTECHNIQUE MONTRÉAL
affiliée à l'Université de Montréal

Cette thèse intitulée :

Scalable Algorithms for Discrete Choice Modeling and Assortment Optimization

présentée par **Claudio SOLE**
en vue de l'obtention du diplôme de *Philosophiæ Doctor*
a été dûment acceptée par le jury d'examen constitué de :

DEDICATION

To my family . . .

ACKNOWLEDGEMENTS

Here I am, writing the acknowledgments of my PhD thesis! The last words about such an incredible, life changing journey of five years. The list of people to thank has grown very long, and starts with my advisors Andrea Lodi and Sanjay Dominik Jena: without your patience and support I would not be here today, writing these acknowledgments. But also, thank you Andrea for giving me the opportunity to come to Montreal and enjoy one of the (or maybe *the*) most significant experiences of my life. All the people I met, everything I learned, everything I saw, starts with that talk in Bologna when I first met you. And thank you Sanjay for all the insightful discussions and for always bringing me back on track whenever I got lost in papers and experiments. Also, I am very grateful for your support during the last year, dealing with all the difficulties related to pandemic and self-isolation.

Thank you to all my other collaborators: Markus Leitner, Behrouz Babaki, Laurent Charlin: it was great working with you. And thank you Roberto Roberti: I was lucky enough to work with you again, after my master, and I am extremely grateful for that.

A huge thank you goes to Mehdi and Mariia for their positivity and willingness to help, and to Koladé and Khalid, always there to support me, to cheer me up, and to offer me sweets for my coffee breaks.

A special thank you goes to my colleagues, the best ones I could hope for: thank you Giulia for being the first person (with Michele) to welcome me in Montreal, and to help me going through the myriad of things I had to do to settle in this city; thank you Mathieu, for all the time you spent satisfying my curiosity and answering my questions, and for your kindness and altruism; thank you Antoine, Gabriele and Federico: your daily presence and our dinners together have been fundamental for my happiness and well being in Montreal; thank you Greta for the great companion you have been since the beginning of the PhD (so many evenings and weekends spent together at the office); thank you Luciano and Jaime for all your help, for the chats about Inter and Serie A, and for being such amazing people; and thank you to all other great people I met in the office: Leandro, Gonzalo, Maxime, Didier, Prateek, Giacomo, I am so grateful for all the time we spent together.

Thank you to all my friends outside of the office: thank you Lucas, Charles, Florian and Rami for all the fun we had together.

Thank you Alexandra, for your love, and for sharing with me some of the toughest periods of my PhD, always reminding me my value, supporting me, and pushing me to have an healthy approach to work and life in general.

Finally, thank you to the most precious people in my life: mamma, papà, Roberto and Federica, whose unconditional love and support help me go through any struggle in life.

RÉSUMÉ

L’optimisation de l’assortiment vise à identifier un ensemble de produits à offrir aux clients, de manière à maximiser le revenu attendu d’une entreprise. L’un des principaux défis à relever pour résoudre cette tâche provient du fait que les demandes de produits sont interdépendantes, en raison des effets dits de substitution et de halo. Par conséquent un modèle de choix discret doit être appris à partir des données recensées, afin de capturer les comportements d’achat des clients confrontés à des ensembles discrets d’alternatives. Un tel modèle peut ensuite être utilisé comme une sous-routine prédictive pour optimiser les décisions d’assortiment. En pratique, la résolution d’un problème d’optimisation de l’assortiment nécessite donc de trouver le bon compromis entre i) la flexibilité du modèle de choix, qui doit être suffisamment général pour bien représenter le comportement d’achat des clients, et ii) la difficulté de résolution du problème d’optimisation pour trouver l’ensemble de produits qui maximise les revenus. Dans cette thèse, nous traitons à la fois les aspects prédictifs et prescriptifs de ce compromis, en fournissant des procédures d’estimation efficaces pour des modèles de choix discrets généraux et des algorithmes pour identifier des assortiments à revenu élevé.

Tout d’abord, nous proposons une procédure efficace pour optimiser les assortiments pour des scénarios dans lesquels chaque produit est associé à un certain coût que l’entreprise doit payer pour offrir ce produit. Nous supposons que les clients choisissent selon un modèle Multinomial Logit. Ce problème est NP-hard. Néanmoins, nos résultats indiquent qu’il est possible d’identifier l’assortiment optimal en une fraction de seconde, en moyenne, même pour des instances avec mille produits. Nous montrons également comment adapter notre approche au cas où les assortiments doivent satisfaire une contrainte sur le nombre maximum de produits disponibles.

Deuxièmement, nous proposons un modèle de choix partiellement rangé, qui généralise les modèles de choix entièrement rangé en supposant que les clients forment des préférences strictes uniquement parmi quelques produits pertinents, tout en étant “indifférents” parmi les autres. Ce modèle permet de capturer tout effet de substitution entre les produits. Nous proposons ensuite une procédure d’estimation basée sur la génération de colonnes pour identifier efficacement les comportements des clients en fonction des données historiques de vente. De plus, nous montrons comment optimiser les décisions d’assortiment basées sur un tel modèle.

Enfin, nous étendons notre modèle de choix partiellement rangé et notre cadre d’estimation pour capturer les comportements d'achat, tels que les effets de halo, qui ne peuvent être expliqués par aucun des modèles appartenant à la famille de la maximisation de l'utilité aléatoire. Nos résultats sur un ensemble de données de ventes d'épicerie dans le monde réel montrent que la prise en compte de ce type de comportements de choix permet d'augmenter de manière significative la précision prédictive.

ABSTRACT

Assortment Optimization aims at identifying a set of products to be offered to customers, so as to maximize a firm's expected revenue. One of the major challenges in solving this task stems from the fact that products demands are interdependent, due to so-called substitution and halo effects. Hence, most often a discrete choice model must be learned from historical data, in order to capture the buying behaviors of customers faced with discrete sets of alternatives. Such model can then be used as a predictive subroutine for optimizing assortment decisions. In practice, tackling the assortment optimization problem thus requires finding the right trade-off between i) flexibility of the choice model, which should be general enough to approximate well the buying behavior of customers, and ii) tractability of the optimization problem one must solve to find the revenue-maximizing set of products. In this thesis, we deal with both the predictive and prescriptive aspects of such trade-off, providing effective estimation procedures for general discrete choice models and scalable algorithms to identify high-revenue assortments.

First, we propose an effective procedure to optimize assortments for scenarios in which each product is associated to a certain cost representing, e.g., the stocking or shipping one, which the firm has to pay for offering that product. We assume customers choose according to a Multinomial Logit model. This problem is NP-hard. Exact solution methods have been shown to be computationally impractical, thus motivating a number of approximate approaches. Notably, our results indicate that it is possible to identify the optimal assortment in a fraction of a second, on average, even on large-scale instances. We further show how to adapt our approach to the case in which assortments must satisfy a constraint on the maximum number of available products.

Second, we propose a partially-ranked choice model that generalizes fully-ranked choice models by assuming that customers may form strict preferences only among few relevant products, while being "indifferent" among the others. Such model can theoretically approximate any model belonging to the *Random Utility Maximization* family, and allows to capture any pattern of substitution among products. We then provide an estimation procedure based on column generation to effectively identify customer behaviors consistent with historical sales data. Further, we show how to optimize assortment decisions based on such model.

Finally, we extend our partially-ranked choice model and estimation framework to capture buying behaviors, such as halo effects, which cannot be explained by any of the models belonging to the *Random Utility Maximization* family. Our results on a real-world grocery sales dataset show that accounting for this type of choice behaviors allows to significantly boost predictive accuracy.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS AND ACRONYMS

AO-B&C	Assortment Optimization via Branch-and-Cut
AO-Boost	Assortment Optimization via Boosting
AOP	Assortment Optimization Problem
AOPC	Assortment Optimization with Product Costs
CDM	Context-Dependent random utility Model
CG	Column Generation
GPT	Growing Preference Tree
GSP	Generalized Stochastic Preference
GT	Ground-Truth
Halo-MNL	Multinomial Logit with Halo effects
IIA	Independence of Irrelevant Alternatives
KL	Kullback-Leibler divergence
LoR	Loss of Rationality
LS	Local Search
MICQ	Mixed-Integer Conic Quadratic
MILP	Mixed-Integer Linear Programming
MIP	Mixed-Integer Programming
MMNL	Mixed Multinomial Logit
MNL	Multinomial Logit
NLP	Non-Linear Programming
PCMC	Pairwise Choice Markov Chain
RB	Rank-Based
RBM	Resiticted Boltzman Machines
RMSE	Root Mean Squared Error
RUM	Random Utility Maximization
SLSQP	Sequential Least Sqares Programming
UPC	Universal Product Code

LIST OF APPENDICES

CHAPTER 1 INTRODUCTION

1.1 Background and Motivation

The problem of identifying a set of high-revenue products to show to customers has become of crucial importance for the success of both brick-and-mortar and online businesses. Apparel retailers, for example, must rearrange the assortments carried in stores due to seasonal trends, while consumer electronics retailers may need to keep up-to-date with customers' changing tastes and new device models. When a customer enters the store and cannot find the product she is looking for, she may react by leaving with no purchase or by substituting the desired product by a similar, available one. Customers' substitution behaviors can be stock-out based, when the considered product is unavailable due to a stock-out event (i.e., the product ran out of availability) or assortment based, when the absence of the product is due to assortment decisions. In general, bad assortment decisions may result in lost sales, added operational costs, and lead to customer insatisfaction. For example, one may be tempted to offer large assortments, with the hope of increasing the probability of customers finding what they are looking for. This, however, may deteriorate customers' experience, due to the cognitive burden of processing too much information, and ultimately jeopardize sales [3, 4]. Also, the presence of low-revenue items may cannibalize the sales of high-revenue ones, thus leading to inferior profit margins. Ignoring this type of interactions, by assuming the demands of products do not depend on the set of available alternatives, may lead to sub-optimal revenues (see, e.g., [5]).

Motivated by such observations, discrete choice models have been applied in the context of demand forecasting in order to capture assortment-dependent effects on product demands. In particular, consider a universe $\mathcal{N} = \{1, \ldots n\}$ of n products. A discrete choice model then consists of a conditional probability distribution $P(i|S)$ that, for a given assortment of products $S \subseteq \mathcal{N}$, yields the probability of a random arriving customer choosing item i out of the available ones. Assuming each product is associated with a certain revenue r_i, for $i \in \mathcal{N}$, one may then leverage such predictive model to "evaluate" a candidate assortment S, in terms of its expected revenue $R(S) = \sum_{i \in S} r_i P(i|S)$. The *optimal* assortment S^* can then be found by solving the following decision problem:

$$S^* = \underset{S \subseteq \mathcal{N}}{\operatorname{argmax}}\, R(S). \tag{1.1}$$

In order to choose the "right" predictive model, one must thus understand how this choice may impact the final assortment decisions. On the one hand, a flexible choice model may be able to capture complex choice behaviors, thus providing accurate demand forecasts, but lead to an intractable formulation of problem (1.1); on the other hand, simple predictive models may be easier to optimize on, but too restrictive in terms of assumptions they make about the choice behavior of customers. This, in turn, may lead to poor demand forecasts, and ultimately to sub-optimal revenues. It is therefore of paramount importance to understand the structural differences among different models of choice, the assumptions about the choice behavior of consumers from which they are derived, and the practical implications of such assumptions.

A brief overview of Discrete Choice Modeling

In order to understand and predict the decision-making outcome of an agent, economists formalized the definition of *rationality*. In principle, such definition should take into account the set of personal tastes and values of the agent under examination, as the scenario under which she is asked to make a decision. However, such an agent- and scenario-specific definition of rationality would be of rather limited practical utility, since her choices would be difficult to relate to other agents and scenarios. Also, many of these pieces of information are not directly observable by researchers, and therefore cannot be taken into account for predicting choice outcomes. Hence, the standard theory of rational choice relies on more general principles, resting upon the view of decision-makers as *utility maximizers*. According to this view, a decision-maker assigns a certain utility to each alternative, representing the overall interest she may have in choosing that option. When faced with a discrete set of alternatives, she picks the one with the highest utility. Formally, this model of decision-making assumes the existence of an utility function $u : \mathcal{N} \to \mathbb{R}$, mapping each item to a one-dimensional "score" value, which establishes a total ordering among alternatives.

The fundamental principle stemming from the utility maximization framework is the one of *preference transitivity*, according to which the relative preference between two items should not depend on the other items available in the assortment. Formally,

$$p(i|S) \geq p(j|S) \implies p(i|S') \geq p(j|S') \quad \forall S, S' \subseteq \mathcal{N}.$$

In particular, let assume item i is preferred to item j on a given assortment $S \subseteq \mathcal{N}$. This implies that $u(i) \geq u(j)$, which must hold for any other assortment $S' \subseteq \mathcal{N}$, with $S' \neq S$. As a consequence, i must be preferred to j in every assortment $S' \neq S$ as well. In the seminal work of Luce [6], an even stronger principle of preference transitivity was proposed, which makes specific assumptions about the choice probabilities of items in different choice scenarios. According to such principle, known as the *Independence of Irrelevant Alternatives* (IIA), not only the relative preference, but also the ratio of the choice probabilities of two items should not change, independently of the other available alternatives. Formally, this implies that

$$\frac{P(i|S)}{P(j|S)} = \frac{P(i|S')}{P(j|S')} \quad \forall S, S' \subseteq \mathcal{N}.$$

In the same work, Luce derived the Multinomial Logit (MNL) model, arguably the most studied choice model in the literature, directly from the implications of the IIA principles on choice probabilities. According to such model, the probability of choosing item j from assortment S is given by

$$P(j|S) = \frac{e^{u_j}}{\sum_{i \in S} e^{u_i}}.$$

In practice, researchers can neither observe the utilities assigned to alternatives by a decision-maker, nor several of the other factors driving her choice. Hence, Marschak [7] introduced in economics the family of *Random Utility Maximization* (RUM) models. These models decompose item-specific utilities U_i in two parts, i.e., $U_i = v_i + \epsilon_i$. The first part, v_i, is the *observed* portion of utility, usually a function of item features and other explanatory variables describing the decision-maker, while ϵ_i is a *random* noise which accounts for unobserved factors influencing the choice. The probability of choosing a certain item $i \in \mathcal{N}$ from assortment S is then given by

$$\begin{aligned} P(i|S) &= P(U_i > U_j) & \forall j \in S \setminus \{i\} \\ &= P(v_i + \epsilon_i > v_j + \epsilon_j) & \forall j \in S \setminus \{i\} \end{aligned}$$

Different assumptions regarding the distribution of the random components lead to different choice models. In particular, Marschak [7] showed that the logit formula can be derived in terms of a RUM model, where the random components follow an extreme value distribution, and are identically and independently distributed among items.

A significant amount of literature was devoted to test whether the IIA principle actually holds in practice, and robust empirical evidence was found of choice scenarios in which IIA may be violated, leading to intransitive preferences. From a cognitive perspective, such violations are induced by the similarity of options, and are thus referred to as *similarity effects*. A famous example in this regard is the one of the red/blue buses (see, e.g., [8]), considering the case of commuters choosing a mode of transportation for traveling to work. Specifically, assume initially that a red bus and a car are available for reaching a certain destination, and that commuters choose either option with 50% probability. In a second moment, a new bus service is made available for the same trip. Such service is identical to the former except for the color of the new bus, which is blue. How will this new option affect commuters' choice? The MNL model will predict an equal share of approximately 33% for all the alternatives,[1] which is counter-intuitive. Likely, commuters that chose the car in the first scenario, will stick to their preference, while bus commuters may equally split among the two buses, therefore violating IIA. This type of choice behaviors is still compatible with the RUM framework, and can be explained by allowing for correlations in the utility-error components of similar alternatives, as in the Nested Logit model [9]. Also, when analyzing choice outcomes from multiple decision-makers, even if each of them does show transitive preferences, *aggregated* choice probabilities may show IIA violations. More powerful choice models, such as the Mixed Multinomial Logit (MMNL) [10, 11] and the Rank-Based [12] are based on this observation and assume the existence of multiple classes of IIA-consistent decision-makers over which choices are aggregated. Notably, these models can approximate any model belonging to the RUM family and capture any type of substitution behavior among products.

However, RUM models do obey to the so-called *Regularity* principle of (rational) choice, according to which the probability of choosing a certain alternative can only decrease when a new alternative is added to the offered ones. Formally,

$$P(i|S') \leq P(i|S) \quad \forall S \subseteq S' \subseteq \mathcal{N}, i \in \mathcal{N}.$$

Choice behaviors violating the Regularity assumption have been widely documented. The work of Simonson and Tversky [13] shows that such violations may be induced by *loss aversion*, a cognitive bias leading to choice phenomena known as *compromise effects*, where middle (i.e., compromise) options in terms of overall evaluation are preferred to extreme ones. In the same work, the compromise effect was shown to cause intransitive preferences as well, therefore violating the IIA principle. Violations of the Regularity assumption may arise also from the introduction in the assortment of an option, referred to as the *decoy option*, whose presence increases the sales of another product perceived as much better on all dimensions. While different cognitive biases lead to different choice phenomena, the literature about choice behaviors in contrast with the regularity assumption generally refers to such behaviors as *product synergies* or *halo effects* (see, e.g., [14, 15]).

[1] In the first scenario we have P(car)/P(red-bus)=1. According to the IIA principle, this must hold in the second scenario as well. Moreover, the two buses will have the same (or very similar) utilities, assuming the bus color does not significantly affect commuters' choice. As a consequence, P(car)=P(red-bus)=P(blue-bus)

In order to visually summarize our discussion above, we report in Figure 1.1 (adapted from [16]) different scenarios in which items are evaluated on two dimensions and, based on their position in such attribute-space, trigger different choice phenomena and substitution behaviors. In particular, items i and z may cannibalize each other sales in the leftmost scenario, given their similarity on both dimensions; item z may be preferred to the other alternatives in the center scenario since considered an acceptable trade-off in terms of both attributes; finally, in the rightmost scenario, item z (i.e., the decoy option) may increase the attractiveness of item i, since it is dominated by the latter on both dimensions.

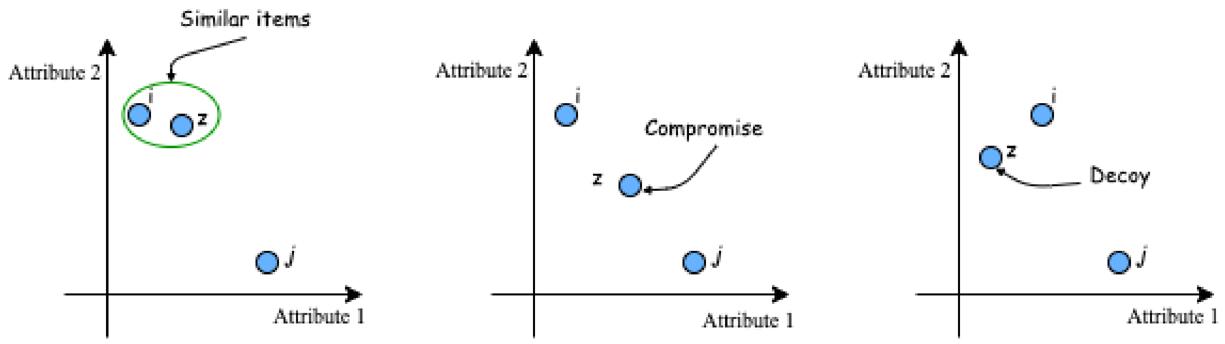


Figure 1.1 Choice phenomena resulting from different evaluations of two-dimensional items: Similarity effect (Left), Compromise effect (Center) and Decoy effect (Right).

We conclude this section by noticing that frameworks of choice other than the RUM one have been proposed in the literature in order to account for choice phenomena such as similarity, compromise and decoy effects. For example, *consider-then-choose* choice models assume agents may fail to maximize their utility payoff due to limited attention. In particular, when entering a store, agents may decide to focus on a small subset of the available items, so as to avoid the cognitive burden of evaluating all of them; such agents may then choose among the considered items only. Context-dependent choice models, on the other hand, explicitly take into account the effects of all available alternatives on item-specific utilities. We refer the reader to Chapter 2 and Section 6.2 for a deeper overview on this stream of literature.

1.2 Objectives and Contributions

In this thesis, we deal with both the predictive and prescriptive aspects of the Assortment Optimization problem (1.1), by focusing on choice models characterized by different levels of flexibility. In particular, as discussed in Section 1.1, different choice models make different assumptions about the buying behavior of customers, which impact both the predictive accuracy of the model and the computational tractability of the resulting decision problem. Their applicability thus depends on the type (and amount) of data available for learning, and the computational requirements that need to be satisfied. Hence, we start by analyzing a variant of the assortment optimization problem under the Multinomial Logit model, still widely adopted in practice despite its limited flexibility. We then increase, in each of the following contributions, the variety of choice behaviors we aim to capture. In particular, our contributions are the following.

An Exact Method for (Constrained) Assortment Optimization Problems with Product Costs: Our first contribution focuses on the problem of Assortment Optimization with Product Costs (AOPC), when customers choose according to a Multinomial Logit model. Product costs allow to model a variety of real-life situations where a firm may incur operational costs (such as stocking or ordering ones) for offering a certain product. Moreover, several solution methods for other problems in the literature of assortment optimization and network revenue management rely on the solution of one or more subproblems taking the form of an AOPC. Although practically relevant, solving the AOPC to optimality is NP-hard [17]. We propose an exact solution method for the AOPC, showing that instances with up to one thousand products can be solved to optimality in less than a second, on average. Moreover, we describe how to adapt the proposed approach to the case with cardinality constraints on the size of the assortment.

A Partially Ranked Choice Model for Large-Scale Data-Driven Assortment Optimization: Our second work contributes to the literature on rank-based choice models, which are able to approximate any model belonging to the RUM family, and thus capture any type of substitution among products. In particular, we propose a discrete choice model based on partially-ranked preferences, which assume that customers have strict preferences only on a small set of relevant products, while being *indifferent* to the others. Besides being a behaviorally plausible model of choice, this model allows for an effective estimation procedure, where customer types are discovered in a data-driven way using a tree-like data structure. Furthermore, we show how to optimize assortments over the proposed choice model. Our numerical results show that the proposed choice model delivers accurate predictions of customers' buying behavior, leading to close-to-optimal assortments at decision time.

On the estimation of discrete choice models to capture irrational customer behaviors: In our final contribution, we show that the estimation framework of partially-ranked preferences is flexible enough to account for the discovery of *irrational*, rank-based behaviors. These, in particular, allow to capture assortment-dependent effects on product demands, which lead to violations of the regularity assumption, thus escaping the limitations of the RUM framework. An extensive set of experiments shows that on synthetic and real-life datasets, accounting for irrational choice behaviors can significantly boost predictive accuracy.

CHAPTER 2 LITERATURE REVIEW

Many planning problems in Operations Management require a prior estimation of the demand of the products or services involved. Originally, such demands were assumed to be independent from each other, resulting in the so-called *indpendent demand model*, according to which a customer either buys her favorite product, or leaves without any purchase. This model may well represent scenarios in which product offerings are highly differentiated and, due to various types of restrictions, targeted to a specific class of customers [18]. Also, Cohen et al. [19] and Cohen et al. [20] empirically observed that cross-item effects (such as substitution and halo effects) may be negligible in those product categories where customers are brand loyal. However, such assumption has been shown to be overly restrictive in general, and unable to capture commonly observed buying behaviors. For example, the studies of [21] and [22] report that 45% and 62% of customers, respectively, may substitute a stocked-out product for an available one. While heuristic corrections to the independent demand model have been proposed to capture such behaviors (see, e.g., [23]), discrete choice models have been widely adopted with the aim of taking customer choice behavior into account when optimizing strategic decisions.

Choice-based demand forecasting The MNL model is arguably the most studied choice model in the literature, thanks to its interpretability and tractable estimation. The seminal work of [24] used an MNL model in the context of brand choice analysis. The works of [25], [5] and [26] use MNL-based models of demand to analyze the impact of choice-based approaches to airline revenue management. In such context, results from [5] indicate that taking into account the choice behavior of customers when optimizing fare classes availability may lead to revenue improvements between 1% and 5%. Recently, Feldman et al. [27] applied the MNL in the frame of online recommender systems. Their results show that such model can outperform state-of-the-art methods from the Machine Learning literature based on the independent demand assumption, in terms of revenue generated from recommended products.

Despite its popularity, the MNL model obeys to the IIA principle, which limits the complexity of substitution behaviors it can capture. Hence, more complex choice models have been designed over the years to overcome its limitations. Among these, the class of *parametric* choice models, to which the MNL belongs, makes *a priori* assumptions about the choice behavior of customers, i.e., before fitting the model to the data. As a consequence, under- and over-fitting issues may arise whenever the assumptions made are not appropriate for the given scenario. The Exponomial [28], the Nested Logit [9] and the Markov Chain [29] models belong to this class, and so does the MMNL, when the number of customer classes is fixed.

Non-parametric choice models, on the other hand, allow to gain structural flexibility as more data becomes available, therefore circumventing the need for extensive model selection. In particular, relatively simple models (i.e, with few parameters) will be fit in limited data-regime settings, therefore limiting the risk of capturing spurious patterns from data; however, as the amount of available data increases, so does the number of parameters in the model and, in consequence, the complexity of choice behaviors it can capture. Motivated by these observations, Farias et al. [12] introduced rank-based choice models in the context of Operations Management. Such models are defined in terms of a probability distribution over customer types. These are represented by rankings over products, such that high-rank products are preferred to low-rank ones. When faced with a set of alternatives, a customer picks the product with the highest rank among the available ones. It follows that the number of possible customer types is factorially large in the number of products. As a consequence, estimating such models is computationally challenging. The work of [30] tackles such difficulties by proposing a column-generation procedure in which the subproblem aims to identify customer types consistent with sales data. However, as pointed out by the authors, the computational burden of their approach grows significantly with the number of available transactions. Moreover, relatively small numbers of products were used in their experiments. The work of [31] proposes to speed-up the solution of the subproblem by means of a local search method. However, its computational complexity is quadratic in the number of products, thus making such approach impractical for large number of products.

All the choice models mentioned above follow the RUM framework. Among these, the Markov chain, the Rank-Based and the MMNL models can theoretically approximate any model belonging to the RUM family [1], and are thus able to capture any pattern of substitution among products. In [32], the authors empirically compare the RUM choice models most common in the literature of assortment optimization, when it comes to predicting the choice behavior of customers on new assortments. Among the tested models, the Exponomial and Markov Chain choice models tend to be the most accurate when little and large amounts of training data are available, respectively. Also, their results confirm that rank-based choice models are relatively data-efficient, exhibiting stable performances independently of the amount of available data.

All the models that belong to the RUM family obey to the regularity assumption. Recently, Jagabathula and Rusmevichientong [33] analyzed a grocery sales dataset, showing a

[1]For the Markov chain choice model, this is not true in general. However, Blanchet et al. [29] show that under some mild conditions, the Markov Chain choice model can provide a relatively good approximation to any RUM model. The approximation is exact for some models, such as the MNL one.

significant presence of choice behaviors incompatible with the RUM framework on several categories of products. As a consequence, RUM choice models may not be able to fit well such datasets, ultimately leading to poor demand forecasts. Several choice models have been proposed in the literature to escape the limitations of the RUM framework (see, also, Section 6.2). Among those, the Halo-MNL and the Context-Dependent random utility Model (CDM) proposed by Maragheh et al. [15] and Seshadri et al. [34], respectively, extend the MNL model in order to capture (positive) pairwise interactions among products. In [35], the authors propose adaptions of rank-based models in order to capture compromise and decoy effects. Their model assumes customers share the same preferences, i.e., ranking over products. In [36], the authors propose to model choice probabilities as given by the stationary distribution of a continuous time Markov chain, whose nodes are indexed by the available alternatives. However, these models do not subsume the RUM family of models, and therefore may not be able to provide a better fit of the data even in the presence of choice behaviors violating the regularity assumption. In this regard, the authors of [33] highlighted the need for more general choice models, subsuming the RUM framework and able to capture choice phenomena such as halo effects. Within the domain of Operations Management, the Generalized Stochastic Preference choice model [2], and the Decision Forest choice model [37, 38] have been recently proposed to address this kind of need. Nevertheless, the former lacks practical estimation methods, and no empirical study has been performed regarding its ability to predict the buying behavior of customers; the latter, while being able to capture *any* type of choice behavior, may not have enough structure to generalize well in limited data-regime settings, besides imposing difficult computational challenges for its estimation. In this regard, Chen and Mišic [37] discuss several strategies to tackle both the predictive and computational aspects in a principled way.

Tractability of the decision problem. An important aspect to take into consideration when modeling demand in the context of assortment optimization, is the tractability of the decision problem under the considered model. Notably, such problem can be solved efficiently under the MNL choice model by means of the *revenue-ordered assortments* greedy algorithm [25] or by linear programming [39]. In the same line, Blanchet et al. [29] and Feldman and Topaloglu [40] provide efficient solution methods for the assortment optimization problem under the Markov Chain choice model.

However, optimizing assortments under several other choice models is generally NP-hard. This has been shown in [14] for the Nested Logit model, in [41, 42] for the MMNL choice model, and in [43, 44] for the case of Rank-Based choice models. Intractability results have been obtained for practically relevant variants of the assortment optimization problems as well, such as when considering product costs under the MNL model [17], when optimizing assortments subject to space constraints under a nested logit model [45] and for the case of assortment optimization with cardinality or capacity constraints, under the Markov Chain model [46]. Motivated by such results, a significant body of literature focused on approximate solution methods, so as to obtain primal and dual bounds to the assortment optimization problem in a tractable manner (see, e.g., [47, 48, 49, 50, 51, 52]).

Exact methods for NP-hard variants of the assortment optimization problem received relatively little attention in the literature. Among them, Bront et al. [41] and Méndez-Díaz et al. [53] provide compact Mixed-Integer Linear Programming (MILP) formulations for the assortment optimization problem under the MMNL choice model. However, such formulations have been shown not to scale well in the number of products. In [54], the authors consider the capacity-constrained variant of the same problem, for which they propose a Mixed-Integer Conic Quadratic (MICQ) formulation. Their approach favourably compares with the MILP formulations previously mentioned in terms of computing times. The works [55, 56] provide MILP formulations for optimizing assortment decisions under Rank-Based choice models. Recently, Alfandari et al. [57] proposed an exact solution method for optimizing assortments under the Nested Logit model.

CHAPTER 3 ORGANIZATION OF THE THESIS

The remainder of this thesis is organized as follows. Chapter 4 describes our work on assortment optimization with product costs. In Chapter 5, we introduce our partially-ranked choice model, together with the estimation procedure we propose and the mathematical formulation used to optimize assortment decisions under this model. In Chapter 6, we describe our estimation procedure for a discrete choice model aiming to capture halo effects. Chapter 7 provides a discussion about the general focus of this thesis, and how each contribution relates to it. Finally, Chapter 8 concludes the thesis, outlines its limitations and presents possible directions for future developments.

CHAPTER 4 ARTICLE 1 - AN EXACT METHOD FOR (CONSTRAINED) ASSORTMENT OPTIMIZATION PROBLEMS WITH PRODUCT COSTS

Authors: Markus Leitner, Andrea Lodi, Roberto Roberti, Claudio Sole
Submitted to *Operations Research*

Abstract We study the problem of optimizing assortment decisions in the presence of product-specific costs when customers choose according to a multinomial logit model. This problem is NP-hard and approximate solutions methods have been proposed in the literature to obtain both primal and dual bounds in a tractable manner. We propose the first exact solution method for this problem and show that provably optimal assortments of instances with up to one thousand products can be found, on average, in about two tenths of a second. In particular, we propose a bounding procedure based on the approximation method of [1] to provide tight primal and dual bounds at a fraction of their computing times. We show how these bounds can be used to effectively identify an optimal assortment. We also describe how to adapt our approach for handling cardinality constraints on the size of the assortment or space/resource capacity constraints.

4.1 Introduction

Composing a set of products that maximizes the (expected) profit is a major planning problem in retail operations, revenue management, and online advertising, see, e.g., Gallego and Topaloglu [58], Kök and Vaidyanathan [23]. The need to consider customer choice behavior in this decision has led to the consideration of a large variety of corresponding models, particularly including parametric discrete-choice models based on random utility theory [59]. This led to a vast body of literature on *assortment optimization problems* (AOP) under variants and extensions of the *multinomial logit model* (MNL) [60]. Despite some drawbacks, such as the assumption of independence of irrelevant alternatives that are overcome by some of its extensions, the MNL remains one of the most frequently studied discrete-choice models in assortment optimization. The success of the MNL is due to the fact that it can be efficiently estimated and its interpretability, which can allow to gain insights into the choice behavior of consumers.

We focus on variants of the *assortment optimization problem with product costs* (AOPC) under the MNL as introduced by Kunnumkal et al. [17] (see, also, Kunnumkal and Martínez-De-Albéniz [51]) who also show that the AOPC is NP-hard. The AOPC is particularly important as it models a variety of practical applications where a certain cost (e.g., stocking or shipping costs) is incurred by the firm for offering a given product. Furthermore, the AOPC appears as a subproblem when solving other problems encountered in assortment optimization and revenue management where business constraints, such as space (see, e.g., [61]) or resource capacity constraints (see, e.g., [62]), must be satisfied, or when optimizing assortment decisions over a mixture of customer types (see, e.g., [1]). In particular, (approximate) solutions schemes based on the Lagrangian relaxation that rely on the solution of AOPs with product costs have been proposed to solve these problems (see, e.g., [1]).

Building upon the parametric dual bound presented by Feldman and Topaloglu [1] and a mixed-integer linear programming (MILP) presented by Kunnumkal and Martínez-De-Albéniz [51], we propose an exact method that can find an optimal assortment for AOPC instances featuring up to 1000 products in about two tenths of a second of computing time on average. To the best of our knowledge, this is the first exact algorithm that is not based on simply casting a MILP model or a mixed-integer conic quadratic (MICQ) model into a general-purpose black-box solver. We show that our method significantly outperforms these models in terms of size of the instances that can be solved and in terms of computing time by orders of magnitude.

The remainder of this article is organized as follows. Section 4.2 summarizes the literature related to the AOPC. Section 4.3 formally introduces the AOPC and reviews two exact formulations from the literature that can be solved with general-purpose black-box solvers. Section 4.4 introduces the exact solution method we propose. Section 4.5 shows how the method can be extended to the constrained AOPC. Computational results are reported in Section 4.6. Some conclusions are drawn in Section 4.7.

4.2 Literature review

In the following, we provide a brief literature review mainly focusing on assortment optimization problems under variants of the MNL that are directly relevant for our developments.

Talluri and van Ryzin [25] show that the AOP under the MNL can be solved in polynomial time by considering revenue-ordered product subsets. While the capacity constrained AOP (under the MNL) can still be solved in polynomial time [63], most other practically interesting problem variants are NP-hard. This includes, in particular, the AOPC studied in [17] and the AOP under the more general mixed-multinomial logit model (MMNL), which considers multiple customer classes [41, 42]. Consequently, a large body of literature is devoted to the study of (polynomial time) approximation algorithms for AOPs, see, e.g., Feldman and

Topaloglu [52], Feldman et al. [64], Gallego and Topaloglu [65], Liu et al. [66] for recent contributions.

Despite being a natural and important generalization, the AOPC has been addressed by few articles only. Kunnumkal et al. [17] and Kunnumkal and Martínez-De-Albéniz [51] reformulate the AOPC as a parametric optimization problem and show how to derive primal and dual bounds, respectively, on the optimal expected profit in polynomial time. The former work proposes a 2-approximation algorithm with complexity $O(n^3)$, where n is number of products considered, and a polynomial time approximation scheme for computing feasible assortments to the AOPC with theoretical guarantees on their expected revenue. Complementary to this work, Kunnumkal and Martínez-De-Albéniz [51] propose a procedure to obtain upper bounds on the optimal revenue that relies on the solution of $O(n^3)$ continuous knapsack problems. Their approach favorably compares against a MILP formulation for the AOPC provided by the authors in the same work (which we will recall in Section 4.3), with relatively tight bounds obtained in short computing times.

The AOPC has also been considered in other AOP variants. Feldman and Topaloglu [1] suggest a solution method based on the Lagrangian relaxation to approximately solve the AOP under the MMNL. They relax the constraints ensuring that the same products are offered to all customer types and therefore obtain the AOPC as subproblem. In order to obtain dual bounds to the AOPC in a tractable manner, they propose a grid-based approach where denser grids provide tighter bounds, at the cost of higher computing times. Feldman and Paul [61] relate the approximability of the space-constrained AOP to the solution of AOPs with fixed costs. Honhon et al. [67] propose polynomial time algorithms for different variants of an AOP with fixed costs under a ranking-based customer choice model. In the context of network revenue management, the column-generation procedure proposed by Kunnumkal and Topaloglu [68] relies on the solution of a subproblem that has the form of an AOPC, and Kunnumkal and Topaloglu [62] propose a Lagrangian-decomposition approach that relies on the solution of a series of AOPs with fixed costs, where the Lagrangian multipliers associated with the relaxed constraints play the role of product fixed costs.

As mentioned above, the literature on exact methods for NP-hard AOPs under variants of the MNL is relatively limited. In this regard, Bront et al. [41] and Méndez-Díaz et al. [53] show that an AOP under the (M)MNL can be reformulated as a MILP with a polynomial number of variables and constraints. Although solving this compact MILP (by a black-box solution framework) does not scale well in practice, such approach is typically used to assess the performance of alternative methods. Şen et al. [54] suggest to reformulate the (constrained) AOP under the MMNL as a conic (quadratic) mixed-integer program that can be solved by black-box solution frameworks too. The results of their computational study indicate clear advantages of their approach over previously considered MILP-based methods. Their formulation will be adapted to the AOPC in Section 4.3 and considered in our computational study. Recently, Alfandari et al. [57] propose an exact algorithm based on fractional programming for the AOP under the nested logit model.

The paper that is closest to ours is the one by Feldman and Topaloglu [1], which we will describe in Section 4.4.1. In particular, one of the contributions of our work is to embed their grid-based approximation method into a bounding procedure, where coarser grids are used to alleviate the computational burden of denser ones, allowing to obtain both primal and dual bounds of the same quality at a small fraction of their computing times. Furthermore, we show how to leverage such bounds to solve the AOPC to optimality.

4.3 Problem description and formulations from the literature

The AOPC can be formally described as follows. A set of n products $P = \{1, 2, \ldots, n\}$ is given. Each product $j \in P$ is characterized by a revenue $r_j > 0$, a product costs $c_j \geq 0$, and a preference weight (or, simply, preference) $v_j > 0$. The preference of not making any purchase is denoted as $v_0 \geq 0$. Let $x_j \in \{0, 1\}$ be a binary decision variable indicating if product $j \in P$ is included in the assortment ($x_j = 1$) or not ($x_j = 0$). For each $\boldsymbol{x} \in \{0, 1\}^n$, let $v(\boldsymbol{x})$ be defined as $v(\boldsymbol{x}) = \sum_{j \in P} v_j x_j$. According to the MNL, the probability that a customer purchases product $j \in P$ is $v_j x_j/(v_0 + v(\boldsymbol{x}))$, and the no-purchase probability is $v_0/(v_0 + v(\boldsymbol{x}))$. The objective of the AOPC is to find a set of products $P^* \subseteq P$ such that the corresponding profit $z(P^*) = \sum_{j \in P^*} r_j v_j/(v_0 + \sum_{j \in P^*} v_j) - \sum_{j \in P^*} c_j$ is maximum. The AOPC can be formulated as the mixed integer non-linear programming problem

$$z^* = \max_{\boldsymbol{x} \in \{0,1\}^n} \left\{ \frac{\sum_{j \in P} r_j v_j x_j}{v_0 + v(\boldsymbol{x})} - \sum_{j \in P} c_j x_j \right\}. \tag{4.1}$$

Two exact reformulations of problem (4.1) that can be cast into general-purpose black-box solvers have been proposed recently. The first formulation is a MILP model provided by Kunnumkal and Martínez-De-Albéniz [51], see also Gallego and Topaloglu [58]. For each $j \in P$, in addition to decision variable $x_j \in \{0, 1\}$, let $u_j \geq \mathbb{R}_+$ be a non-negative continuous variable representing the purchasing probability of product $j \in P$, i.e., $u_j = v_j x_j/(v_0 + v(\boldsymbol{x}))$. Similarly, let $u_0 \geq \mathbb{R}_+$ be a continuous variable representing the no-purchase probability, i.e.,

$u_0 = v_0/(v_0 + v(\boldsymbol{x}))$. The MILP provided by Kunnumkal and Martínez-De-Albéniz [51] is

$$z^* = \max \sum_{j\in P} (r_j u_j - c_j x_j) \tag{4.2a}$$

$$\text{s.t. } v_0 u_j \leq v_j u_0 \qquad \forall j \in P \tag{4.2b}$$

$$u_j \leq \frac{v_j}{v_0 + v_j} x_j \qquad \forall j \in P \tag{4.2c}$$

$$u_0 + \sum_{j\in P} u_j = 1 \tag{4.2d}$$

$$x_j \in \{0,1\} \qquad \forall j \in P \tag{4.2e}$$

$$u_j \in \mathbb{R}_+ \qquad \forall j \in P \cup \{0\} \tag{4.2f}$$

The objective function (4.2a) maximizes the profit of the selected products. Constraints (4.2b) ensure that the ratio between the purchasing probabilities of product $j \in P$ and the no-purchase probability is consistent with the corresponding preferences v_j and v_0. Constraints (4.2c) force the purchasing probabilities of all products not included in the assortment to be zero. Constraint (4.2d) makes sure that the sum of purchasing probabilities is equal to one. Constraints (4.2e)-(4.2f) define the domain of the variables.

The second formulation is based on a MICQ proposed by Şen et al. [54] for the AOP under the MMNL. In particular, we adjust the corresponding objective function to the AOPC by adding a costs-related term, which penalizes the introduction of products in the assortment. Specifically, let $\bar{r}$ be the maximum revenue over all products, i.e., $\bar{r} = \max\{r_j \,|\, j \in P\}$. let $\varphi_j \in \mathbb{R}_+$ be a non-negative continuous variable equal to $1/(v_0 + v(\boldsymbol{x}))$ if product $j \in P \cup \{0\}$ is in the assortment and 0 otherwise - notice that φ_0 is equal to $1/(v_0 + v(\boldsymbol{x}))$. Moreover, let $w \in \mathbb{R}_+$ be a non-negative continuous variable equal to the sum of the preferences of the selected products including the no-purchase preference. The AOPC can be formulated as

$$
\begin{aligned}
z^* = \ & \overline{r} - \min\Big(\overline{r}v_0\varphi_0 + \sum_{j\in P} v_j(\overline{r} - r_j)\varphi_j + \sum_{j\in P} c_j x_j\Big) && && \text{(4.3a)}\\
\text{s.t.}\ & w = v_0 + v(\boldsymbol{x}) && && \text{(4.3b)}\\
& \varphi_j w \geq x_j^2 && \forall j \in P && \text{(4.3c)}\\
& \varphi_0 w \geq 1 && && \text{(4.3d)}\\
& v_0\varphi_0 + \sum_{j\in P} v_j\varphi_j \geq 1 && && \text{(4.3e)}\\
& x_j \in \{0,1\} && \forall j \in P && \text{(4.3f)}\\
& \varphi_j \in \mathbb{R}_+ && \forall j \in P \cup \{0\} && \text{(4.3g)}\\
& w \in \mathbb{R}_+ && && \text{(4.3h)}
\end{aligned}
$$

The objective function (4.3a) aims at maximizing (or equivalently, minimizing the negative) expected profit. Here, the first three terms denote the revenue of the selected products, while last term accounts for the total cost of adding such products to the assortment. Constraint (4.3b) sets variable w equal to the sum of the preferences of the selected products plus the no-purchase preference. Constraints (4.3c) guarantee that $\varphi_j = 1/w$ if product $j \in P$ is selected and 0 otherwise, since the problem is in minimization form and variables φ_j have non-negative objective coefficients. Constraint (4.3d) sets φ_0 equal to $1/w$. Constraint (4.3e) corresponds to (4.2d) and is redundant but helps significantly strengthen the linear relaxation of (4.3) (see Şen et al. [54]). Constraints (4.3f)-(4.3h) define the domain of the variables.

Şen et al. [54] also show how to strengthen the linear relaxation of formulation (4.3) by adding the following McCormick inequalities [69]:

$$
\begin{aligned}
& m_j^1 x_j \leq \varphi_j \leq M_j^1 x_j && \forall j \in P && \text{(4.4a)}\\
& \varphi_0 - M_j^0(1 - x_j) \leq \varphi_j \leq \varphi_0 - m_j^0(1 - x_j) && \forall j \in P && \text{(4.4b)}
\end{aligned}
$$

where m_j^1 and M_j^1 are appropriate lower and upper bounds on the value of variable φ_j when product $j \in P$ is selected, M_j^0 and m_j^0 are an upper and a lower bound on the value of variable φ_0 when product $j \in P$ is not selected. For the unconstrained AOPC, m_j^1 and M_j^1 are set equal to $1/(v_0 + \sum_{i\in P} v_i)$ and $1/(v_0 + v_j)$, respectively, whereas M_j^0 and m_j^0 are set equal to $1/(v_0 + \sum_{i\in P\setminus\{j\}} v_i)$ and $1/v_0$, respectively. For the constrained AOPC (i.e., when cardinality or budget constraints are present), tighter values of parameters m_j^1, M_j^1, M_j^0, and m_j^0 can be computed by solving some binary (multiple) knapsack problems or the corresponding linear relaxations (see Şen et al. [54]).

4.4 Exact solution method

In this section, we describe our exact method, which builds upon the parametric dual bound proposed by Feldman and Topaloglu [1] (summarized in Section 4.4.1). The exact method is based on three main ideas. The first idea (see Section 4.4.2) is to enhance the bounding procedure of Feldman and Topaloglu [1] by limiting the number of parametric bounds to compute and sequentially improve the global dual bound with the goal of finding a small range for the sum of the preferences of the products of any optimal assortment. The second idea (see Section 4.4.3) is to use the primal and dual bounds computed by the bounding procedure to rule out the products that cannot be part of any optimal assortment. The third idea (see Section 4.4.4) is to find an optimal assortment by solving problem (4.2), with a general-purpose solver, and add some constraints to limit the search within the range of preferences returned by the bounding procedure.

4.4.1 The approximation method of Feldman and Topaloglu [1]

Feldman and Topaloglu [1] propose the following approximation method that computes (provably) tight dual bounds to the AOPC. Their method is based on formulating the AOPC as

$$z^* = \max_{p \in [p_{\min}, 1]} \{z(p)\} \tag{4.5}$$

where p is the no-purchase probability and $p_{\min}$ is the minimum no-purchase probability (achieved when all products are in the assortment). By normalizing the purchase preferences such that $v_0 = 1$ (which can be done w.l.o.g.), these values are computed as $p = 1/(1 + v(\boldsymbol{x}))$ and $p_{\min} = 1/(1+\sum_{j \in P} v_j)$. Furthermore, $z(p)$ is the optimal value of the parametric problem

$$z(p) = \max_{\boldsymbol{x} \in \{0,1\}^n} \left\{ \sum_{j \in P} (p r_j v_j - c_j)\, x_j \;\middle|\; \frac{1}{1 + v(\boldsymbol{x})} = p \right\}, \tag{4.6}$$

which can be equivalently written as (see Feldman and Topaloglu [1])

$$z(p) = \max_{\boldsymbol{x} \in \{0,1\}^n} \left\{ \sum_{j \in P} (p r_j v_j - c_j)\, x_j \;\middle|\; v(\boldsymbol{x}) \leq \frac{1}{p} - 1 \right\}.$$

Solving (4.5) by computing $z(p)$ is intractable because it requires solving a binary knapsack problem, which is NP-hard, for each value of $p \in [p_{\min}, 1]$. The parametric problem (4.6) can, however, be adjusted to compute provably tight dual bounds to z^* as follows.

For any pair of values $\underline{p}, \overline{p} \in [p_{\min}, 1]$ such that $\underline{p} \leq \overline{p}$, let $\overline{z}(\underline{p}, \overline{p})$ be a parametric dual bound to $\max_{p \in [\underline{p}, \overline{p}]} z(p)$ (i.e., $\overline{z}(\underline{p}, \overline{p}) \geq \max_{p \in [\underline{p}, \overline{p}]} z(p)$) defined as

$$\overline{z}(\underline{p}, \overline{p}) = \max_{\boldsymbol{x} \in [0,1]^n} \left\{ \sum_{j \in P} (\overline{p} r_j v_j - c_j)\, x_j \,|\, v(\boldsymbol{x}) \leq \frac{1}{\underline{p}} - 1 \right\}. \tag{4.7}$$

Feldman and Topaloglu [1] show that

$$z(G) = \max_{k \in \{1,\ldots,K\}} \left\{ \overline{z}(p^k, p^{k+1}) \right\} \tag{4.8}$$

is a valid dual bound to z^* for any set of grid points $G = \{p^k \,|\, k \in \{1, \ldots, K+1\}\}$, such that $p_{\min} = p^1 \leq p^2 \leq \ldots \leq p^K \leq p^{K+1} = 1$, which can be computed by solving K continuous knapsack problems. Each of these knapsack problems can be solved by considering and adding (if feasible) the products $j \in P$ in non-increasing order of their utility-to-space consumption ratios $(\overline{p} r_j v_j - c_j)/v_j$.

The gap between $z(G)$ and z^* is due to two sources of *error*. The first source is that z^* is computed over all values of p in the interval $[p_{\min}, 1]$ whereas $z(G)$ is computed over a set of grid points and p, in (4.6), is replaced by $\underline{p}$ and $\overline{p}$ in the objective function and the right-hand side of the constraint of (4.7), respectively. Hence, denser grids provide better approximations of z^*. The second source is that the decision variables to compute $\overline{z}(\underline{p}, \overline{p})$ are in the range $[0,1]^n$ whereas the AOPC imposes the integrality constraints on each x_j, $j \in P$; nevertheless, the linear relaxation of the binary knapsack problem usually provides tight dual bounds.

Feldman and Topaloglu [1] discuss properties of effective grids and illustrate the benefits of using an exponential grid. For a fixed parameter $\rho > 0$ (e.g., $\rho = 0.1, 0.01, 0.001, \ldots$), the exponential grid $G_\rho^{\exp}$ is defined as $G_\rho^{\exp} = \{(1+\rho)^{-k+1} \,|\, k \in \{1, \ldots, K(\rho)+1\}\}$, where $K(\rho)$ is such that $(1+\rho)^{-K(\rho)} \leq p_{\min} < (1+\rho)^{-K(\rho)+1}$. The exponential grid of points $G_\rho^{\exp}$ clearly covers the entire interval $[p_{\min}, 1]$. Feldman and Topaloglu [1] show that $z(G_\rho^{\exp})$ cannot be improved by more than a factor $1+\rho$ by any other grid. Since $p_{\min} < (1+\rho)^{-K(\rho)+1}$, we have $K(\rho) = O(-\log(p_{\min})/\log(1+\rho))$. Therefore, for example, if $p_{\min} = 0.25$ and we would like to have a performance guarantee of 0.1%, we can choose $\rho = 0.001$, which implies $K(\rho) = 1387$. In other words, tight dual bounds to z^* can be achieved by using exponential grids with a relatively small number of points.

4.4.2 Enhanced bounding procedure

We now discuss how $z(G_\rho^{\exp})$ can be computed also for very dense exponential grids by sequentially computing dual bounds over grids with increasing density.

For each pair of values $\underline{p}, \overline{p} \in [p_{\min}, 1]$ such that $\underline{p} \leq \overline{p}$, let $\underline{z}(\underline{p}, \overline{p})$ be a primal bound to z^*. We compute this primal bound by considering the products $j \in P$ in non-increasing order of their utility-to-space consumption ratio $(\overline{p} r_j v_j - c_j)/v_j$ and adding a given product to the assortment $P' \subseteq P$ as long as the sum of the preferences of the products in P' does not exceed $\frac{1}{\underline{p}} - 1$. This simple heuristic for the binary knapsack problem, which usually provides tight primal bounds, yields an assortment P' with a profit of $\underline{z}(\underline{p}, \overline{p}) = \sum_{j \in P'} (p r_j v_j - c_j)$, where $p = 1/(1 + \sum_{j \in P'} v_j)$.

For a given $G_\rho^{\exp}$ and the corresponding parametric bounds $\underline{z}(p^k, p^{k+1})$, $\overline{z}(p^k, p^{k+1})$ (for $k \in \{1, \dots, K(\rho)\}$), let

$$lb(G_\rho^{\exp}) = \max_{k \in \{1, \dots, K(\rho)\}} \{\underline{z}(p^k, p^{k+1})\} \tag{4.9}$$

be the best primal bound to z^* computed over $G_\rho^{\exp}$. Moreover, let $\mathcal{I}(G_\rho^{\exp})$ be the set of intervals, defined by $G_\rho^{\exp}$, that contains a value p for which $z(p)$ is the optimal solution of (4.5). Set $\mathcal{I}(G_\rho^{\exp})$ is defined as

$$\mathcal{I}(G_\rho^{\exp}) = \{[p^k, p^{k+1}] \mid k \in \{1, \dots, K(\rho)\} \,:\, \overline{z}(p^k, p^{k+1}) \geq lb(G_\rho^{\exp})\}. \tag{4.10}$$

Let $\mathcal{P}(G_\rho^{\exp})$ be the union of all intervals of $\mathcal{I}(G_\rho^{\exp})$, i.e., $\mathcal{P}(G_\rho^{\exp}) = \cup_{[p^k, p^{k+1}] \in \mathcal{I}(G_\rho^{\exp})} [p^k, p^{k+1}]$. We will refer to the minimum and maximum values of $\mathcal{P}(G_\rho^{\exp})$ as $p_{\min}(G_\rho^{\exp})$ and $p_{\max}(G_\rho^{\exp})$, respectively, i.e., $p_{\min}(G_\rho^{\exp}) = \min\{p \,|\, p \in \mathcal{P}(G_\rho^{\exp})\}$ and $p_{\max}(G_\rho^{\exp}) = \max\{p \,|\, p \in \mathcal{P}(G_\rho^{\exp})\}$.

As $lb(G_\rho^{\exp})$ is a primal bound to z^* and $\overline{z}(p^k, p^{k+1})$ is a dual bound to $z(p)$ with $p \in [p^k, p^{k+1}]$, $k \in \{1, \dots, K(\rho)\}$, we can observe that, for any optimal AOPC solution P^*, there must exist an interval $[\underline{p}, \overline{p}] \in \mathcal{I}(G_\rho^{\exp})$ such that $p^* \in [\underline{p}, \overline{p}]$, where $p^* = 1/(1 + \sum_{j \in P^*} v_j)$, so p^* belongs to $\mathcal{P}(G_\rho^{\exp})$.

This observation suggests that we can compute $z(G_\rho^{\exp})$ even for highly dense grids (e.g., defined with $\rho = 1\text{E}{-7}$), by exploiting sets $\mathcal{I}(G_\rho^{\exp})$ computed with much higher values of ρ (e.g., $\rho = 1\text{E}{-2}$ or $1\text{E}{-3}$). In particular, given ρ' and ρ such that $\rho' < \rho$, $z(G_\rho^{\exp})$ can be computed as

$$z(G_\rho^{\exp}) = \max_{k \in \{1, \dots, K(\rho)\} \,:\, [p^k, p^{k+1}] \cap \mathcal{P}(G_{\rho'}^{\exp}) \neq \varnothing} \{\overline{z}(p^k, p^{k+1})\}. \tag{4.11}$$

We propose a dual bounding procedure that computes a sequence of dual bounds to z^* of increasing tightness. Given two parameters ρ^f and ρ^ℓ (the first and the last value of ρ – e.g.,

$\rho^f = 1\text{E}{-2}$ and $\rho^\ell = 1\text{E}{-7}$), the procedure first computes $z(G_\rho^{\text{exp}})$ according to (4.8) with $\rho = \rho^f$, and then sequentially computes $z(G_\rho^{\text{exp}})$ according to (4.11) for $\rho = 0.1\rho^f, 0.01\rho^f, \dots, \rho^\ell$ and by using $\mathcal{P}(G_{\rho'}^{\text{exp}})$ with $\rho' = \rho^f, 0.1\rho^f, \dots$ In the computational results of Section 4.6, we show that the number of intervals for which the parametric bound (4.7) must be computed with this procedure is significantly lower than by computing $z(G_\rho^{\text{exp}})$ as in (4.8), with $\rho = \rho^\ell$.

4.4.3 Variable fixing to rule out products

The bounding procedure described in Section 4.4.2 can be speed up by ruling out some products that cannot be part of an optimal assortment. For each value of $\rho = \rho^f, 0.1\rho^f, \dots, \rho^\ell$, the bounding procedure returns primal and dual bounds ($lb(G_\rho^{\text{exp}})$ and $z(G_\rho^{\text{exp}})$) to z^* and the set $\mathcal{P}(G_\rho^{\text{exp}})$, which allow to fix the variable x_j of some products $j \in P$ to 0 according to the following propositions.

Proposition 1 *For any ρ, each product $j \in P$ such that $p_{max}(G_\rho^{exp}) r_j v_j - c_j < 0$ cannot be part of an optimal assortment.*

Proof. By definition, $p_{\max}(G_\rho^{\text{exp}})$ is greater than or equal to the no-purchase probability of any optimal assortment. If $p_{\max}(G_\rho^{\text{exp}}) r_j v_j - c_j < 0$ for a given product $j \in P$, then $p r_j v_j - c_j$ is negative for any $p \in \mathcal{P}(G_\rho^{\text{exp}})$. Therefore, such product cannot yield a positive profit and cannot be part of any optimal assortment. □

For ease of notation, let $\widetilde{p}_j^{k+1} = p^{k+1} r_j v_j - c_j$ denote the profit of product $j \in P$ in the interval $[p^k, p^{k+1}] \in \mathcal{I}(G_\rho^{\text{exp}})$. As previously mentioned, computing the parametric bound $\overline{z}(p^k, p^{k+1})$ requires sorting the products by non-increasing utility-to-space consumption ratios $\widetilde{p}_j^{k+1}/v_j$ and filling up the knapsack starting from a product with the largest ratio. For each interval $[p^k, p^{k+1}]$, we can thus keep track of the critical product $s(k)$, i.e., the largest index such that $\sum_{j=1}^{s(k)-1} v_j \le 1/p^k - 1$. We can then state the following variable-fixing criterion.

Proposition 2 *For a given ρ, each product $j \in P$ such that, for all intervals $[p^k, p^{k+1}] \in \mathcal{I}(G_\rho^{exp})$, the following two conditions hold*

$$(i)\ \ \widetilde{p}_j^{k+1}/v_j < \widetilde{p}_{s(k)}^{k+1}/v_{s(k)} \quad \textit{and} \quad (ii)\ \ \overline{z}(p^k, p^{k+1}) + \widetilde{p}_j^{k+1} - v_j \widetilde{p}_{s(k)}^{k+1}/v_{s(k)} < lb(G_\rho^{exp})$$

cannot be part of any optimal assortment.

Proof. Let $\overline{x}$ denote the optimal solution of problem (4.7) with objective value $\overline{z}(p^k, p^{k+1})$, for a given $[p^k, p^{k+1}] \in \mathcal{I}(G_\rho^{\text{exp}})$. We observe that $\overline{x}_j = 0$ for each product $j \in P$ satisfying the first condition (since its utility-to-space ratio is smaller than the one of the critical product $s(k)$)

and that the quantity $\widetilde{p}_j^{k+1} - v_j \widetilde{p}_{s(k)}^{k+1} / v_{s(k)}$ is a lower bound of the profit decrease obtained by setting $\overline{x}_j = 1$ (see, e.g., [70], Section 2.2.3). Hence, $\overline{z}(p^k, p^{k+1}) + \widetilde{p}_j^{k+1} - v_j \widetilde{p}_{s(k)}^{k+1} / v_{s(k)}$ is a valid upper bound to the profit $\overline{z}(p^k, p^{k+1})_{|x_j=1}$ of any solution (of the given interval $[p^k, p^{k+1}] \in \mathcal{I}(G_\rho^{\exp})$) containing product j. The proposition follows by noticing that if $\overline{z}(p^k, p^{k+1})_{|x_j=1} < lb(G_\rho^{\exp})$ for all $[p^k, p^{k+1}] \in \mathcal{I}(G_\rho^{\exp})$, i.e., if introducing product j in any of the solutions corresponding to $[p^k, p^{k+1}] \in \mathcal{I}(G_\rho^{\exp})$, results in a profit lower than the best-known feasible solution, then product j cannot be part of any optimal assortment. $\square$

4.4.4 Finding an optimal assortment

The bounding procedure described in Section 4.4.2 returns two important values, namely $p_{\min}(G_\rho^{\exp})$ and $p_{\max}(G_\rho^{\exp})$, for $\rho = \rho^\ell$, that identify a crucial feature of any optimal assortment: the corresponding no-purchase probability lies in the range $[p_{\min}(G_\rho^{\exp}), p_{\max}(G_\rho^{\exp})]$. To identify an optimal assortment, we solve problem (4.2), with a general-purpose solver, with the addition of the constraints

$$p_{\min}(G_\rho^{\exp}) \leq u_0 \leq p_{\max}(G_\rho^{\exp}), \tag{4.12}$$

which limit the search for an optimal assortment to assortments having no-purchase probability within the range $[p_{\min}(G_\rho^{\exp}), p_{\max}(G_\rho^{\exp})]$.

In the following, we refer to problem (4.2) plus constraints (4.12) as MILP+.

4.5 Constrained AOPC

In real-life applications, there could be additional constraints that should be taken into account when selecting an assortment. The two most common examples of such constraints are *cardinality constraints* and *space/resource constraints* (see, e.g., [1, 61, 51]). Here, we show how the exact method described in Section 4.4 can be adjusted to handle a cardinality constraint. A space/resource capacity constraint can be handled similarly. A cardinality constraint implies that no more than a given number of κ products can be included in any assortment, i.e.,

$$\sum_{j \in P} x_j \leq \kappa \ . \tag{4.13}$$

The following three modifications are made to the exact method to consider constraint (4.13):

1. The presence of constraint (4.13) requires solving the linear relaxation of a two-constrained binary knapsack problem (see [71, 72]) when computing the parametric dual bound

$\overline{z}(p^k, p^{k+1})$. This value can be computed efficiently in polynomial time $O(n^2)$ by dualizing constraint (4.13) through a non-negative Lagrangian multiplier $\lambda \in \mathbb{R}_+$ (see [71]). We exploit the similarity of the two-constraints binary knapsack problems whose linear relaxations need to be solved to compute $z(G_\rho^{\exp})$ (i.e., K problems in total) by calculating an upper bound $\overline{z}(p^k, p^{k+1}, \lambda)$ to $\overline{z}(p^k, p^{k+1})$ using the following procedure. For a given grid density ρ, we initialize λ to 0. Starting from the interval corresponding to $p^k \leq p_{\max}(G_\rho^{\exp}) \leq p^{k+1}$, we compute a "good" Lagrangian multiplier (and upper bound) iteratively by using, for iteration t, $\lambda_t = \lambda_{t-1} + \delta$, with δ small enough. We stop the procedure at iteration $\bar{t}$ if the upper bound starts deteriorating, i.e., if $\overline{z}(p^k, p^{k+1}, \lambda_{\bar{t}}) > \overline{z}(p^k, p^{k+1}, \lambda_{\bar{t}-1})$. We then use the close-to-optimal multiplier $\lambda_{\bar{t}}$ to initialize the procedure for the next interval, thus avoiding starting from scratch, i.e., by assuming $\lambda = 0$. The same approach is used for all other intervals, i.e., until $p^k \leq p_{\min}(G_\rho^{\exp}) \leq p^{k+1}$. Experimentally, we found this procedure to recover good multipliers and upper bounds in a quite small number of iterations.

2. Products are added to assortment P' as long as the sum of the preferences of the selected products does not exceed $\frac{1}{\underline{p}} - 1$ *and* if the total number of products P' is not greater than κ when computing $\underline{z}(p^k, p^{k+1})$ (see Section 4.4.2).

3. Constraint (4.13) is added to MILP+ when computing an optimal assortment (see Section 4.4.4).

4.6 Computational results

In this section, we first describe the instances used to test our exact method (see Section 4.6.1). Then, we report the results achieved by our exact method on the AOPC (see Section 4.6.2) and the cardinality-constrained AOPC (see Section 4.6.3). We also compare the results of our exact method with those of the two formulations from the literature described in Section 4.3.

4.6.1 Test instances

To test our exact method, we generate a set of 800 instances as described by Kunnumkal and Martínez-De-Albéniz [51]. Each instance has $n \in \{100, 200, 500, 1000\}$ products. The preference v_j of product $j \in P$ is computed as $v_j = w_j / \sum_{k=1}^{n} w_k$, where w_j is uniformly distributed in the interval $(0, 1]$. The no-purchase probability is set equal to $v_0 = \frac{\Phi}{1-\Phi} \sum_{j \in P} v_j$, where Φ is a parameter in the set $\Phi \in \{0.25, 0.75\}$, meaning that the no-purchase probability is either 25% or 75%. The revenue r_j of product $j \in P$ is sampled from the uniform distribution

$[0, 2000]$, and the product cost c_j is sampled from the uniform distribution $[0, \gamma r_j v_j/(v_0+v_j)]$, where γ is another parameter in the set $\gamma \in \{0.5, 1.0\}$. Therefore, we have 16 combinations of instances, (n, Φ, γ). For each combination, we generate 50 test instances.

4.6.2 Computational results on the AOPC

Table 4.1 summarizes the computational results achieved with our exact method (hereafter, called LLRS) on the 800 test instances and compares its performance with the performance of MILP (4.2) and MICQ (4.3) both solved with Cplex 20.1. All experiments are conduced on a single core of a machine with 500GB-RAM and an Intel(R) Xeon(R)Gold 6142 with 2.60GHz CPU. A time limit of ten minutes is imposed on each experiment.

Our exact method has two parameters: ρ^f and ρ^ℓ. After some parameter tuning, we set these parameters as $\rho^f = 1\text{E}{-2}$ and $\rho^\ell = 1\text{E}{-7}$. Our implementation has been coded in C++ and compiled with `gcc 4.8.5`; Cplex 20.1 is used to solve problem MILP+.

For each of the three methods and each combination of test instances, Table 4.1 reports the number of instances solved to optimality (opt), the average final gap (gap) in percentage between the best primal and dual bounds found by the corresponding method, the average computing time ($\mathsf{cpu}_{\mathsf{avg}}$), and the maximum computing time ($\mathsf{cpu}_{\mathsf{max}}$).

Table 4.1 shows that LLRS outperforms the two formulations from the literature in terms of both number of instances solved to optimality and computing time. Indeed, LLRS can solve all 800 instances whereas the MILP (4.2) and the MICQ (4.3) can solve 355 and 197 instances only, respectively. The average computing time of LLRS is at least four/five orders of magnitude lower than the computing time of the other two methods. We can observe that the average computing time of LLRS increases with the number of products and that instances featuring lower product costs (i.e., $\gamma = 0.5$) or lower no-purchase probability (i.e., $\Phi = 0.25$) are more difficult to solve than instances with higher product costs (i.e., $\gamma = 1.0$) or higher no-purchase probability (i.e., $\Phi = 0.75$). It is interesting to observe that the average computing time of LLRS on the large instances with 1000 products is about 0.2 seconds and the maximum computing time is just 1.78 seconds.

Table 4.2 reports additional information on the performance of LLRS for each combination of test instances. For the bounding procedure described in Section 4.4.2, Table 4.2 reports the final average dual gap ($\mathsf{gap}_{\mathsf{dual}}$) in percentage (i.e., the gap between z^* and $z(G_\rho^{\exp})$ with $\rho = \rho^\ell$), the final average primal gap ($\mathsf{gap}_{\mathsf{prim}}$) in percentage (i.e., the gap between z^* and $lb(G_\rho^{\exp})$ with $\rho = \rho^\ell$), the number of times $lb(G_\rho^{\exp})$ corresponds to the optimal solution ($\mathsf{opt}_{\mathsf{prim}}$), and the average computing time ($\mathsf{cpu}_{\mathsf{avg}}$). For the variable fixing described in

Table 4.1 Summary of the computational results of LLRS on the AOPC and comparison with MILP (4.2) and MICQ (4.3)

	MILP (4.2)				MICQ (4.3)				LLRS		
(n, Φ, γ)	opt	gap	cpu_{avg}	cpu_{max}	opt	gap	cpu_{avg}	cpu_{max}	opt	cpu_{avg}	cpu_{max}
(100, 0.25, 0.5)	48	0.08	112.46	600.00	14	2.20	493.46	600.00	50	0.02	0.10
(100, 0.25, 1.0)	50	0.00	1.40	4.90	43	0.35	185.84	600.00	50	0.01	0.08
(100, 0.75, 0.5)	50	0.00	0.20	0.32	50	0.00	2.28	10.46	50	0.00	0.02
(100, 0.75, 1.0)	50	0.00	0.29	0.80	48	0.03	38.22	600.00	50	0.00	0.02
(200, 0.25, 0.5)	0	11.86	600.00	600.00	0	11.02	600.00	600.00	50	0.02	0.09
(200, 0.25, 1.0)	10	3.83	546.22	600.00	0	9.39	600.00	600.00	50	0.01	0.06
(200, 0.75, 0.5)	50	0.00	0.33	0.53	42	0.04	161.54	600.00	50	0.01	0.04
(200, 0.75, 1.0)	47	0.01	82.21	600.00	0	1.58	600.00	600.00	50	0.01	0.04
(500, 0.25, 0.5)	0	24.84	600.00	600.00	0	25.20	600.00	600.00	50	0.06	0.49
(500, 0.25, 1.0)	0	20.69	600.00	600.00	0	26.38	600.00	600.00	50	0.04	0.19
(500, 0.75, 0.5)	50	0.00	18.25	196.54	0	0.54	600.00	600.00	50	0.05	0.10
(500, 0.75, 1.0)	0	1.34	600.00	600.00	0	3.66	600.00	600.00	50	0.04	0.09
(1000, 0.25, 0.5)	0	28.39	600.00	600.00	0	25.74	600.00	600.00	50	0.29	1.78
(1000, 0.25, 1.0)	0	26.69	600.00	600.00	0	39.00	600.00	600.00	50	0.15	1.06
(1000, 0.75, 0.5)	0	0.07	600.00	600.00	0	0.92	600.00	600.00	50	0.16	0.24
(1000, 0.75, 1.0)	0	2.15	600.00	600.00	0	4.33	600.00	600.00	50	0.12	0.22
Sum	355				197				800		
Avg		7.50	347.58			9.40	467.58			0.06	
Max		28.39		600.00		39.00		600.00			1.78

Section 4.4.3, #out indicates the average number of products ruled out and %out is the number of products ruled out in percentage. Finally, for the resolution of MILP+ (see Section 4.4.4), cpu_{avg} is the average computing time.

Table 4.2 shows that both the primal and dual bounds computed by the bounding procedure are very tight and the primal bound $lb(G_{\rho}^{\text{exp}})$ is most of the times an optimal assortment (in all but 12 instances). We can also observe that roughly half of the computing time of LLRS is spent on the bounding procedure and the other half on solving MILP+. The proposed variable fixing rules allow to rule out about 31% of the products on average.

We conduct another set of experiments to show the importance of applying the bounding procedure described in Section 4.4.2. In particular, we compare the results achieved by setting ρ^f either to 1E−2 or to 1E−7. When ρ^f = 1E−2, we have the results already reported in

Table 4.2 Additional information on the performance of LLRS on the AOPC

	Bounding Procedure				Var Fixing		MILP+
(n, Φ, γ)	gap_{dual}	gap_{prim}	opt_{prim}	cpu_{avg}	#out	%out	cpu_{avg}
(100, 0.25, 0.5)	0.0031	0.0000	49	0.02	62.2	62.2	0.00
(100, 0.25, 1.0)	0.0047	0.0001	49	0.01	72.4	72.4	0.00
(100, 0.75, 0.5)	0.0000	0.0000	50	0.00	0.9	0.9	0.00
(100, 0.75, 1.0)	0.0002	0.0000	50	0.00	14.7	14.7	0.00
(200, 0.25, 0.5)	0.0007	0.0001	48	0.02	113.6	56.8	0.00
(200, 0.25, 1.0)	0.0008	0.0000	49	0.01	137.9	68.9	0.00
(200, 0.75, 0.5)	0.0000	0.0000	50	0.00	0.0	0.0	0.01
(200, 0.75, 1.0)	0.0001	0.0000	50	0.00	26.5	13.3	0.01
(500, 0.25, 0.5)	0.0001	0.0000	49	0.03	221.6	44.3	0.03
(500, 0.25, 1.0)	0.0002	0.0000	47	0.02	304.1	60.8	0.02
(500, 0.75, 0.5)	0.0000	0.0000	50	0.01	0.0	0.0	0.04
(500, 0.75, 1.0)	0.0000	0.0000	50	0.01	65.5	13.1	0.03
(1000, 0.25, 0.5)	0.0000	0.0000	48	0.12	234.6	23.5	0.17
(1000, 0.25, 1.0)	0.0001	0.0000	49	0.06	483.3	48.3	0.09
(1000, 0.75, 0.5)	0.0000	0.0000	50	0.04	0.0	0.0	0.12
(1000, 0.75, 1.0)	0.0000	0.0000	50	0.04	135.0	13.5	0.08
Avg	0.0006	0.0000		0.03	117.0	30.8	0.04

Table 4.1. When $\rho^f = 1\text{E}{-7}$, the bounding procedure computes a single dual bound $z(G_\rho^{\exp})$, with $\rho = 1\text{E}{-7}$, and cannot benefit from the bounds achieved with higher values of ρ; in other words, this approach corresponds to the one proposed by Feldman and Topaloglu [1]. Table 4.3 reports the average and maximum number of intervals for which $\overline{z}(p^k, p^{k+1})$ must be computed ($\#\mathsf{int}_{\mathsf{avg}}$, $\#\mathsf{int}_{\mathsf{max}}$ - notice that $\#\mathsf{int}_{\mathsf{avg}}$ is equal to $\#\mathsf{int}_{\mathsf{max}}$ when $\rho^f = 1\text{E}{-7}$), the average and maximum computing time to solve the problem ($\mathsf{cpu}_{\mathsf{avg}}$, $\mathsf{cpu}_{\mathsf{max}}$), the average and maximum number of intervals, in percentage, for which $\overline{z}(p^k, p^{k+1})$ is computed when $\rho^f = 1\text{E}{-2}$ compared to when $\rho^f = 1\text{E}{-7}$.

Table 4.3 shows that applying the bounding procedure as described in Section 4.4.2 allows to significantly reduce the number of times the parametric bound must be computed: on average by 99.8%. This translates into much lower computing times, which decrease by two/three orders of magnitude on average. We can also observe that $\#\mathsf{int}_{\mathsf{avg}}$ gradually decreases when the number of products increases and, not surprisingly, when the no-purchase option is higher.

Table 4.3 Comparison with the results achieved by LLRS by setting $\rho^f = 1\text{E}{-7}$

	LLRS ($\rho^f = 1\text{E}-7$)			LLRS ($\rho^f = 1\text{E}-2$)					
(n, Φ, γ)	$\#\text{int}_{\text{avg}}$	cpu_{avg}	cpu_{max}	$\#\text{int}_{\text{avg}}$	$\%\text{int}_{\text{avg}}$	$\#\text{int}_{\text{max}}$	$\%\text{int}_{\text{max}}$	cpu_{avg}	cpu_{max}
(100, 0.25, 0.5)	13862947	7.04	7.19	62251	0.4	299206	2.2	0.02	0.10
(100, 0.25, 1.0)	13862947	7.36	7.52	57430	0.4	348920	2.5	0.01	0.08
(100, 0.75, 0.5)	2876823	1.44	1.47	2906	0.1	30049	1.0	0.00	0.02
(100, 0.75, 1.0)	2876823	1.46	1.49	4657	0.2	40371	1.4	0.00	0.02
(200, 0.25, 0.5)	13862947	13.24	13.58	37499	0.3	142049	1.0	0.02	0.09
(200, 0.25, 1.0)	13862947	13.86	14.18	32151	0.2	154331	1.1	0.01	0.06
(200, 0.75, 0.5)	2876823	2.71	2.80	4421	0.2	29177	1.0	0.01	0.04
(200, 0.75, 1.0)	2876823	2.73	2.78	5871	0.2	30551	1.1	0.01	0.04
(500, 0.25, 0.5)	13862947	40.00	42.14	23993	0.2	64459	0.5	0.06	0.49
(500, 0.25, 1.0)	13862947	42.62	43.88	22651	0.2	72100	0.5	0.04	0.19
(500, 0.75, 0.5)	2876823	7.40	7.51	5028	0.2	21586	0.8	0.05	0.10
(500, 0.75, 1.0)	2876823	7.79	8.16	5649	0.2	15801	0.5	0.04	0.09
(1000, 0.25, 0.5)	13862947	91.33	93.87	22701	0.2	42799	0.3	0.29	1.78
(1000, 0.25, 1.0)	13862947	98.27	100.98	19037	0.1	46461	0.3	0.15	1.06
(1000, 0.75, 0.5)	2876823	16.79	17.08	5749	0.2	16638	0.6	0.16	0.24
(1000, 0.75, 1.0)	2876823	17.93	18.29	6076	0.2	16971	0.6	0.12	0.22
Avg	8369885	23.25		19879	0.2			0.06	
Max			100.98			348920	2.5		1.78

4.6.3 Computational results on the cardinality-constrained AOPC

In this section, we report on the performance of LLRS on the cardinality-constrained AOPC. We use the same 800 instances as in the computational study on the AOPC and set the maximum cardinality of the assortments equal to half of the number of products, i.e., $\kappa = n/2$, as done in [51]. For LLRS, parameter δ is set equal to 1E−5.

Table 4.4 summarizes the results achieved on these 800 instances by LLRS in the same format of Table 4.1 and compares them to the results of MILP (4.2) and MICQ (4.3) both solved with Cplex 20.1.

Table 4.4 shows that LLRS outperforms the two formulations from the literature as it can solve all of the 800 instances to optimality, compared to 302 instances solved by the MILP and 354 instances solved by the MICQ, and is orders of magnitude faster than these two formulations. As observed on the AOPC, the computing time of LLRS increases with the

Table 4.4 Summary of the computational results of LLRS on the cardinality-constrained AOPC and comparison with MILP (4.2) and MICQ (4.3)

	MILP (4.2)				MICQ (4.3)				LLRS		
(n, Φ, γ)	opt	gap	cpu_{avg}	cpu_{max}	opt	gap	cpu_{avg}	cpu_{max}	opt	cpu_{avg}	cpu_{max}
(100, 0.25, 0.5)	48	0.08	115.41	600.00	50	0.00	27.15	250.47	50	0.02	0.09
(100, 0.25, 1.0)	50	0.00	1.50	5.03	49	0.05	62.04	600.00	50	0.01	0.08
(100, 0.75, 0.5)	50	0.00	0.39	1.25	50	0.00	0.39	0.87	50	0.08	0.12
(100, 0.75, 1.0)	50	0.00	0.40	0.84	50	0.00	0.94	2.92	50	0.02	0.05
(200, 0.25, 0.5)	0	11.95	600.00	600.00	2	1.68	586.84	600.00	50	0.03	0.10
(200, 0.25, 1.0)	9	3.82	550.88	600.00	1	4.81	595.49	600.00	50	0.01	0.06
(200, 0.75, 0.5)	48	0.00	115.14	600.00	50	0.00	1.28	10.72	50	0.17	0.24
(200, 0.75, 1.0)	47	0.02	81.94	600.00	50	0.00	25.76	299.30	50	0.04	0.08
(500, 0.25, 0.5)	0	24.89	600.00	600.00	0	5.23	600.00	600.00	50	0.08	0.51
(500, 0.25, 1.0)	0	20.75	600.00	600.00	0	10.99	600.00	600.00	50	0.04	0.19
(500, 0.75, 0.5)	0	0.49	600.00	600.00	49	0.00	34.28	600.00	50	0.61	0.74
(500, 0.75, 1.0)	0	1.35	600.00	600.00	2	0.38	584.65	600.00	50	0.13	0.18
(1000, 0.25, 0.5)	0	28.40	600.00	600.00	0	6.27	600.00	600.00	50	0.33	1.96
(1000, 0.25, 1.0)	0	26.72	600.00	600.00	0	14.38	600.00	600.00	50	0.16	1.10
(1000, 0.75, 0.5)	0	0.79	600.00	600.00	1	0.03	598.81	600.00	50	1.83	2.27
(1000, 0.75, 1.0)	0	2.16	600.00	600.00	0	0.62	600.00	600.00	50	0.37	0.54
Sum	302				354				800		
Avg		7.59	391.60			2.78	344.85			0.24	
Max		28.40		600.00		14.38		600.00			2.27

number of products. Furthermore, instances featuring lower product costs (i.e., $\gamma = 0.5$) or higher no-purchase probability (i.e., $\Phi = 0.75$) take more time to solve to optimality. It is interesting to observe that the average computing time of LLRS on large instances with 1000 products is approximately 0.67 seconds, and the maximum computing time over all instances is just 2.27 seconds.

4.7 Conclusions

In this work, we have proposed an exact method for the assortment optimization problem with product costs when customers choose according to a multinomial logit model. This problem is (practically) relevant for several reasons. Indeed, product costs emerge in a variety of real-life scenarios, representing operational costs incurred by the firm for offering a certain product. Furthermore, despite its limitations, the multinomial logit model is one of the most studied discrete choice models, both in academia and industry, due to its interpretability and to the fact that it can be efficiently estimated. Moreover, as observed in a number of other works, the assortment optimization problem with product costs appears as a subproblem when optimizing decisions in the context of assortment optimization and network revenue management problems with side constraints. Kunnumkal et al. [17] showed that solving this problem to optimality is NP-hard, motivating a number of approximate approaches to obtain primal and dual bounds in a tractable manner. We have tackled this problem by proposing a bounding procedure, which builds upon the grid-based approximate approach of Feldman and Topaloglu [1]. In particular, we have exploited primal and dual bounds obtained from coarser grids which are computationally cheap to compute and alleviate the computational burden of denser grids. As a result, the iterative algorithm proposed in this article is able to compute tight primal and dual bounds in a fraction of a second, on instances with up to one thousand products. Furthermore, such instances are solved to optimality in roughly two tenths of a second, on average, by exploiting the obtained bounds. As a final contribution, we show that our approach can be readily adapted to handle a cardinality constraint on the size of assortments, with a limited impact on the computing times.

Future work may include the development of exact algorithms for assortment optimization problems with further side constraints or for problems in which customers choose according to other variants of the multinomial logit model. The algorithm proposed in this article can be a key ingredient of such methods as the assortment optimization problem with product costs appears as a subproblem in several assortment optimization and network revenue management problems.

CHAPTER 5 ARTICLE 2 - A PARTIALLY-RANKED CHOICE MODEL FOR LARGE-SCALE DATA-DRIVEN ASSORTMENT OPTIMIZATION

Authors: Sanjay Dominik Jena, Andrea Lodi, Hugo Palmer, Claudio Sole
Published in *INFORMS Journal on Optimization* [73]

Abstract The assortment of products carried by a store has a crucial impact on its success. However, finding the right mix of products to attract a large portion of the customers is a challenging task. Several mathematical models have been proposed to optimize assortments. Most of them are based on discrete choice models that represent the buying behavior of the customers. Among them, rank-based choice models have been acknowledged for representing well high-dimensional product substitution effects, and therefore reflect customer preferences in a reasonably realistic manner. In this work, we extend the concept of (strictly) fully-ranked choice models to models with partial ranking that additionally allow for indifference among subsets of products, i.e., on which the customer does not have a strict preference. We show that partially-ranked choice models are theoretically equivalent to fully-ranked choice models. We then propose an embedded column generation procedure to efficiently estimate partially-ranked choice models from historical transaction and assortment data. The subproblems involved can be efficiently solved by using a growing preference tree that represents partially-ranked preferences, enabling us to learn preferences and optimize assortments for thousands of products. Computational experiments on artificially generated data and a case study on real industrial retail data suggest that our proposed methods outperform existing algorithms in terms of scalability, prediction accuracy, and quality of the obtained assortments.

5.1 Introduction

Assortment planning denotes the process of identifying the set of products that should be offered to the customers. The planning problem is of paramount importance in operations and revenue management, since the choice of the assortment directly impacts the success of the business. However, from a managerial perspective, identifying the ideal assortment is a difficult challenge. While offering more products to the customer may eventually increase the number of sold items (also referred to as conversion), it is well known that an assortment that is too large may jeopardize the total sales. Of course, limitations of space and capacity may naturally limit the number of products the client will be exposed to. However, one may still observe that the presence of a certain product may decrease the sales of another (which is known as product “cannibalization”). In the same way, the absence of a product may encourage the customer to substitute to another (potentially more profitable) product, illustrating the complex synergies among the products in an assortment.

The problem of finding the optimal assortment is crucial in several different domains. In particular, it is omnipresent in online advertisement on the internet, where it is necessary to decide for the limited number of advertisements that can be shown to a specific user profile in order to maximize the likelihood of conversion. In brick-and-mortar retail, assortment decisions are even more impactful, since the assortments are exposed to several different customer profiles and cannot be personalized to each customer type. Furthermore, changes in those assortments can be longsome and costly, given that products will have to be physically removed from the store and inventory, and typically be liquidated at a far lower price. Mathematical models to help finding the “optimal" assortment are therefore becoming increasingly popular.

Defining a customer choice model that explains the market buying behavior sufficiently well is an essential step to optimize an assortment. Among the vast variety of choice models, rank-based choice models are rapidly gaining popularity. Rank-based choice models represent customer types via ranked preference lists on the available products. Those models hold a few important advantages. They can be easily interpreted by store managers and allow for insights regarding customer segmentation. Furthermore, they can be estimated in a purely data-driven manner without any assumptions on the market structure. In most of the application domains (such as online stores and even retail outlets), collecting transaction and assortment data has become a standard, therefore making such data sufficiently available. Data-driven models that automatically make assortment recommendations based on historical data and with limited user input are likely to dominate the operational planning of the retail industry in the future. However, in practice, those models suffer from several challenges. On the computational side, rank-based choice models are hard to estimate. On the managerial side, even though preference lists (representing different customer types) provide store managers with certain insights, those lists typically contain unnecessarily large numbers (if not all) of products, which limits the usefulness of those lists to understand customer segmentation.

Contributions. In this paper, we introduce a new representation for rank-based choice models that conceptually subsumes classical, fully-ranked models (see, e.g., Farias et al. [12], Bertsimas and Mišic [74]). Next to strictly ranked products, our choice model allows customers to be *indifferent* to a subset of the remaining products and therefore buy any of those products with equal probability if their strictly preferred products are unavailable.

We prove that both fully- and partially-ranked choice models can represent the same buying behavior. However, several fully-ranked preference lists are required to represent a partially-ranked list. We propose an efficient method based on column generation to estimate this choice model by iteratively expanding a preference tree in which each node implicitly represents a partially-ranked customer behavior. While, in theory, finding new columns with negative reduced costs may require the solution of a mixed-integer linear programming problem (MIP), the proposed tree structure allowed us to find those columns trivially throughout all of our computational experiments. The estimation method is computationally attractive, handling instances with large number of products. It is also appealing to store managers, since the generated customer behaviors contain a significantly smaller list of ranked products that are necessary to explain the sales. An existing MIP formulation for assortment optimization is adapted to our partially-ranked choice model, which allows the modeler to easily integrate side-constraints such as capacities and requirements for different product categories (e.g., product subset or precedence constraints). We present extensive numerical results via simulation for choice model estimation and assortment optimization and compare with two recent benchmarks: the column-generation procedure of Bertsimas and Mišic [74] and the k-deletion heuristic proposed by Jagabathula and Rusmevichientong [33]. Finally, we report on a case study with real-world data from a major North-American fashion retail chain, provided by our industrial partner JDA Labs [75].

Organization of the paper. Section 5.2 reviews the literature that is relevant to our work. Section 5.3 introduces the new choice model and the procedure to efficiently identify relevant customer behaviors and the underlying distribution. Section 5.4 provides an MIP formulation to provide an optimized assortment based on our choice model. Section 5.5 reports on numerical experiments from a simulation study on synthetic data, and for real-world data from the retail industry. We conclude in Section 5.6. Formal proofs, further theoretical developments and additional computational results can be found in the appendix and the online supplement of this paper.

5.2 Relevant Literature

Most of the literature acknowledges that practically-effective assortment optimization requires an appropriate choice model that represents the buying behavior of the customers when faced with an assortment. A vast variety of choice models has been applied in different domains such as transportation, marketing, and revenue management. We here focus on those that are most relevant to our work, and refer the reader interested in broad overviews on assortment planning and choice models to surveys such as those of Mahajan and Van Ryzin [76] and Kök et al. [77]. In the context of assortment optimization, two families of choice models have found predominant popularity in the literature and have been successfully applied in the aforementioned domains.

Parametric choice models. The first family is that of (parametric) random utility maximization models. Among its most prominent members is the Multinomial Logit (MNL) choice model, which attributes an utility value to each of the available options. As in most of the choice models in revenue management, customers may also choose to abandon the buying process (e.g., if they are not willing to buy any of the available options), which is typically modeled as a dummy no-purchase option. Even though these models are analytically and computationally tractable, they have several shortfalls. In particular, the MNL assumes the Independence of Irrelevant Alternatives (IIA) property [78], due to which substitution effects among products (such as cannabilization) cannot be captured. Nested logit models, pioneered by Ben-Akiva [79] for the modeling of travel demand, capture certain cases of substitution, but are still subject to the IIA property within each nest. Further extensions, such as Mixed Multinomial Logit (MMNL) models overcome those shortfalls and can capture quite general customer behaviors. Unfortunately, these models are computationally challenging for practical assortment optimization, given that they do not only involve discrete variables, but are also typically non-linear and non-convex. Most importantly, parametric choice models generally rely on a good knowledge of the market structure and do strongly depend on the application context (see, e.g., Jagabathula [80]), which makes them sensitive to issues of under- and over-fitting.

Non-parametric choice models. The second family of choice models are non-parametric exponential models. Among them, rank-based models are quickly gaining popularity in the literature (see, e.g., Jagabathula [80], Van Ryzin and Vulcano [81], Vulcano and Van Ryzin [82]). Rank-based choice models assume that a customer behavior can be represented by a sorted preference list σ of the available options. The customer will then select the option that is ranked highest in her preference list and available in the assortment. Part of the model is a distribution over all possible preference sequences that specifies the probability that a random customer follows a specific buying behavior σ. Rank-based choice models have been acknowledged to offer manifold advantages. They implicitly capture high-dimensional substitution effects and therefore complex synergies among products. They do not make assumptions on the market structure and do therefore not involve the same risks of over- or under-fitting, as it may be the case for parametric models. Further, their distribution can, theoretically, be derived from historical data. Such a purely data-driven approach is attractive from the manager perspective, as it requires little application-specific user input (often none at all) and is therefore easy to apply and to maintain. Having identified the relevant preference sequences may also give managers valuable insights about the customer segmentation.

Unfortunately, learning those choice-models over the space of different preference sequences, which is factorially large in the number of products N, is a major challenge. Both the identification of relevant customer behaviors and computing the underlying discrete *probability mass function (pmf)* that would explain the observed transactions are therefore computationally difficult. In this regard, Jagabathula [80] and Farias et al. [12] strongly advanced research by circumventing the need for searching in factorially large space, without the requirement of a preinformed market structure. These authors consider only those models that minimize the revenue for a given assortment, therefore enabling the use of the dual problem and resulting in the choice model for worst-case prediction. However, while the approach proposed by these authors may yield a good estimate of the worst-case revenue, it may not be ideal for managers, given that the assortment planning based on a worst-case choice model is likely to have a suboptimal performance on average.

Both Haensel and Koole [83] and Vulcano and Van Ryzin [82] focus on the case where the set of different customer types is limited and known beforehand. Instead of aggregating demand data over long periods, they divide the purchase horizon into smaller time periods. Haensel and Koole [83] allow for multiple transactions per period. In contrast, Vulcano and Van Ryzin [82] define at most one transaction per period. Both works explicitly handle data incompleteness that arises from the impossibility of distinguishing a period without a customer and a period with a non-purchasing customer. The former authors estimate those customer arrivals in a pre-processing step, while the latter authors propose an expectation-maximization approach that integrates the estimation of the customer arrival rate. While the assumption of knowing the customer sequences may seem overly restrictive in many settings, the proposed methods can be used in a more complex framework. In this spirit, Van Ryzin and Vulcano [81] proposed the *market discovery algorithm* that generates new customer sequences in a column-generation framework, and estimate the corresponding probability distribution with the aforementioned method [82]. The iterative framework assumes that a restricted Master problem estimates the corresponding *pmf* for a subset of preference sequences, while new preference sequences are generated in a sub-problem.

Bertsimas and Mišic [74] followed a similar approach. The authors work on time-aggregated data and minimize the absolute ℓ_1 error between predicted and historical sales probabilities instead of maximizing the likelihood probability. The final choice model is then used in a novel mixed-integer programming model to provide an optimal assortment. More recently, this assortment optimization formulation has been further explored and improved [56], to speed up the exact solution to the assortment optimization problem. Even though the proposed assortment optimization formulation scales fairly well, the computational complexity of identifying relevant customer behaviors is rather high. The entire process from choice model estimation to the optimized assortment is therefore limited to rather small numbers of products.

Our work is closely related to the aforementioned works (i.e., [81, 74]), as it adapts the general column-generation framework. As Bertsimas and Mišic [74], we focus on the case of aggregated data. However, we do not make any assumptions about the loss function used to estimate the *pmf* other than convexity. Further, Van Ryzin and Vulcano [81] use an integer programming heuristic to identify new reduced costs columns and emphasize that the computational burden increases significantly with the size of the problem, which is given by the number of transactions and is (in practice) directly related to the number of products. Their numerical experiments include not more than 15 products. Our paper explicitly focuses on the efficient estimation of the model when the number of products is large.

Finally, it is worth noting that Ho-Nguyen and Kilinc-Karzan [84] recently provided a uniform view of the methods discussed above. Their method based on saddle point duality estimates rank-based choice models in the context of dynamic learning, when choice models are updated with new data along time. Even though the authors provide a theoretical convergence guarantee for their algorithms, they do not present any numerical experiments.

Partial orders of preferences. Most of the works on rank-based choice models assume that preference sequences must contain (almost) all products. However, it is known that only the first $\bar{n}+1$ ranked items are required to determine a customer choice (see, e.g., Honhon et al. [67]), when the size of the assortments does not exceed $N-\bar{n}$ items (where N is the total number of existing products and $\bar{n}$ is the number of items that are not part of the assortment). The *k-deletion* heuristic [33] exploits this fact by enumerating all possible $O(N^{\bar{n}})$ sequences, which can only be deployed when $\bar{n}$ is small. In contrast, the number of strictly ranked products in our approach is determined dynamically, driven by the data, and can vary among customer behaviors. Further, throughout our computational experiments, our approach generally strictly ranked much less than $\bar{n}+1$ products.

Estimating fully (or almost fully) ranked preference sequences accurately is even more challenging when data on complete ranks is not available. Establishing partial orders among items has been of interest in both the machine learning and the operations management communities. Lebanon and Mao [85] propose a general taxonomy to represent partial preference relationships and estimate the corresponding Mallows model. The model may consist of any sequence of single items or sets of items, where items within the same set are not preferred to each other. While the preference information could essentially be represented by a (possibly quite large) set of pair-wise preferences among items, the proposed partial rankings may allow for a much more compact representation. Jagabathula and Vulcano [86] collect partially-ranked information to construct a directed acyclic graph of preferences for each store customer. They then sample only those fully ranked sequences that are coherent with the preference graph. In this paper, we consider partially ranked sequences whose representation formally falls into the taxonomy of Lebanon and Mao [85]: a list of strictly ranked items followed by sets of items among which no strict preference is established. However, a key difference is that we enforce a uniform probability distribution among the items which are not strictly ranked, allowing us to obtain choice models that can be efficiently estimated and yield computationally tractable models to optimize the final assortment.

5.3 A Partially-Ranked Choice Model

In this section, we will introduce a new representation for rank-based choice models that allows for strict preference on a subset of the products. This representation allows us to efficiently represent and construct relevant customer preferences using a preference tree. In Section 5.3.1, we will first introduce the new choice model and its representation as a tree. Then, in Section 5.3.2, column generation will be used to grow the decision-tree and generate customer preferences.

Notation. We generally use bold faced characters such as $\boldsymbol{x} \in \mathbb{R}^N$ and $\boldsymbol{A} \in \mathbb{R}^{M \times N}$ to represent vectors and matrices. We use x_i to denote the i-th element of vector $\boldsymbol{x}$.

5.3.1 The Choice Model

Rank-based choice models, such as the one used by Farias et al. [12], assume that the market buying behavior is classified into different customer behaviors. Each customer behavior is represented by an ordered list that establishes a strict preference among all products. Preferred products are said to have *high ranks*, while less preferred products are said to have *low ranks*. The customer, following a specific behavior, buys exactly one product, which is the one that is *highest* ranked in her preference list and available in the presented assortment. Consider the set of products $\mathcal{N} = \{1, 2, \ldots, N\}$ and assume further that the customer always has the choice of selecting the no-purchase option 0, which would make her leave the store without any purchase. Following the notation used by Farias et al. [12], we denote a customer behavior by σ and the rank of product i by $\sigma(i) \geq 0$. A fully-ranked customer behavior can be defined as follows.

Definition 1 ***(Fully-ranked customer behavior):*** *A customer behavior σ with a fully-ranked preference sequence may be written as a permutation of all $N+1$ items in $\mathcal{N} \cup \{0\}$.*

As an example, consider 6 available products. The fully-ranked customer behavior $(3,4,1,2,5,0,6)$ indicates that the preferred product is 3, the second preferred product is 4, etc. A customer with such a preference will buy the highest ranked product that is available in the assortment. Note that product 6 will never be bought, since it is ranked after the no-purchase option 0.

A rank-based choice model $(\boldsymbol{\sigma}, \boldsymbol{\lambda})$ is then defined as a set $\boldsymbol{\sigma} = \{\sigma_1, \ldots, \sigma_K\}$ of customer behaviors, together with a probability mass function $\boldsymbol{\lambda} \in \mathbb{R}^K$ that represents the corresponding probabilities that a random customer entering the store follows the specific behavior.

Even though rank-based choice models have several desirable properties, fully-ranked customer preferences are prone to some disadvantages. Their generation is computationally expensive and, from a management perspective, a fully-ranked sequence allows for little insights regarding customer segmentation. This is due to the fact that most of the ranked products explain only marginal portions of the sales (see Observation 1 further below), preventing store managers from identifying those products that actually have high impact (and explain sales). Moreover, a general customer may not have a strict preference order in mind that is defined on all products. Instead, she may rather have a strict preference on a few products, but if none of those is available, she would choose any product from a subset of the available products with similar characteristics. As an example, consider a customer looking for a specific sport shoes model. If this model is not available in the exposed assortment, the customer may choose any other sport shoes model that has similar characteristics and is available in the assortment. Clearly, only a subset of the products available in the assortment complies with these characteristics.

Partially-ranked customer behaviors. The choice model proposed here assumes that customer preference lists do not necessarily have to impose a strict order on all products (which may be in the order of hundreds or thousands). Instead, a customer may have a strict preference on a subset of those products, $P(\sigma)$, e.g., 3, 4 and 1. If those products are absent in the assortment, the customer may buy any similar and available product, e.g., 2, 5 and 6. We may represent such a choice behavior as $\sigma = (P(\sigma), I(\sigma)) = (3,4,1,\{2,5,6\},0)$, where $P(\sigma) = (3,4,1) \subseteq \mathcal{N} \cup \{0\}$ is a strictly ranked list of preferred products and $I(\sigma) = \{2,5,6\} \subseteq \mathcal{N} \cup \{0\} \backslash P(\sigma)$ is a subset of *indifferent* products that will be chosen with uniform probability. There may be more than 6 products, but, assuming that product 0 is either in $P(\sigma)$, in $I(\sigma)$ or after $I(\sigma)$, those products will never be selected.

Definition 2 ***(Simple partially-ranked customer behavior):*** *A simple customer behavior σ contains a strictly ranked preference list $P(\sigma)$ and an indifference set $I(\sigma)$, where $P(\sigma) \subseteq \mathcal{N} \cup \{0\}$ and $I(\sigma) \subseteq \mathcal{N} \cup \{0\} \backslash P(\sigma)$ are mutually exclusive subsets of $\mathcal{N} \cup \{0\}$. Given an assortment S, a customer following behavior $\sigma = (P(\sigma), I(\sigma))$, will select the product that is ranked highest in $P(\sigma)$ and available in S. If $P(\sigma) \cap S = \varnothing$, but $I(\sigma) \cap S \neq \varnothing$, then the customer will select any of the products in $I(\sigma) \cap S$ (which may include the no-purchase option 0) with uniform probability. If, instead, $I(\sigma) \cap S = \varnothing$, the customer does not purchase any item.*

While fully-ranked customer preferences require a hierarchy among all products such that $\sigma(i) < \sigma(j)$ whenever product i is preferred to j, partially-ranked choice models also allow for relations of the form $\sigma(i) = \sigma(j)$, indicating that products i and j are equally preferred and therefore part of the same indifference set. In the example above, the product ranks are as follows: $\sigma(3) = 0$, $\sigma(4) = 1$, $\sigma(1) = 2$, $\sigma(2) = \sigma(5) = \sigma(6) = 3$ and $\sigma(0) = 4$. Again, note that it is not required to list all N products in $P(\sigma)$ or $I(\sigma)$, meaning that $P(\sigma) \cup I(\sigma) \subseteq \mathcal{N} \cup \{0\}$ and the inclusion can be strict. Each product from $\mathcal{N}$ can be part either of the strictly ranked list or of the indifference set, but not in both. Even though Definition 2 assumes that the no-purchase option 0 is always available, all developments made in this paper also apply to the case where 0 is unavailable, assuming that customers always purchase a product. The position of 0 also has implications on possible alternative notations. Specifically, a partially-ranked behavior $\sigma = (P(\sigma), I(\sigma))$ can be equivalently written as $(P(\sigma))$, if $0 \in P(\sigma)$, and as $(P(\sigma), I(\sigma), 0)$, if $0 \notin P(\sigma) \cup I(\sigma)$.

When $P(\sigma)$ and $I(\sigma)$ do not include all products of $\mathcal{N}$, our notion of a partially-ranked customer behavior becomes similar to the one of *consideration sets* (see, e.g., Aouad et al. [87], Jagabathula and Vulcano [86]). However, we further distinguish products in those consideration sets into strictly ranked products and indifferent products, and impose a uniform probability distribution on the latter. Also note that a simple partially-ranked behavior corresponds to the special case $\mathfrak{S}_{1,\ldots,1,k}\pi$ within the taxonomy of Lebanon and Mao [85]: a sequence of strictly ordered items followed by k items among which no strict preference is imposed. However, the simple partially-ranked behavior further assumes a uniform probability distribution among the k items.

Before elaborating on the equivalence between fully- and partially-ranked choice models, we note that the we may further generalize the simple partially-ranked customer behavior to a more complex one, using q alternating lists of preferred products $P^{\ell}(\sigma)$ $(\ell = 1, \ldots, q)$ and indifference sets $I^{\ell}(\sigma)$. This more general behavior type cannot be fit into the taxonomy of Lebanon and Mao [85]. Given that the methodology proposed in this paper is based on simple partially-ranked behaviors, we refer the reader interested in the more general behavior type to Appendix A.1.1.

Equivalence between representations. Even though the partially-ranked choice model seems to be more general than fully-ranked choice models, the underlying preference behaviors are essentially equivalent. Consider a simple partially-ranked customer behavior $(3, \{2, 5, 6\}, 0)$. The same choice model can be represented by a set of fully-ranked preferences with one complete (fully-ranked) list for each of the permutations of the products in the indifference set: $(3, 2, 5, 6, 0)$, $(3, 2, 6, 5, 0)$, $(3, 5, 2, 6, 0)$, $(3, 5, 6, 2, 0)$, $(3, 6, 2, 5, 0)$ and $(3, 6, 5, 2, 0)$. This transformation holds for any simple partially-ranked preference list.

Lemma 1 *Consider a partially-ranked preference list $\sigma_p = (P(\sigma), I(\sigma))$ (see Definition 2), occurring with probability λ_p. We can generate from σ_p a set with $|I(\sigma_p)|!$ fully-ranked preference lists and define the corresponding probabilities such that given the same assortment S, the final probability that product i is bought is the same in both cases.*

A formal proof of Lemma 1 can be found in Appendix A.1.4. We note here that, in practice, it may not be necessary to enumerate all $|I(\sigma_p)|!$ fully-ranked preference lists to represent choice probabilities that are equivalent to σ_p. We elaborate more on this topic in online supplement: Appendix B.1.1. Based on Lemma 1, we now show that both the partially-ranked choice model and the fully-ranked choice model can represent equivalent buying behaviors.

Theorem 1 ***(Equivalence between fully-ranked and partially-ranked choice models):*** *A choice model $(\boldsymbol{\sigma}^C, \boldsymbol{\lambda}^C)$ based on fully-ranked customer behaviors (see Definition 1) can be represented by a choice model $(\boldsymbol{\sigma}^P, \boldsymbol{\lambda}^P)$ that contains only partially-ranked customer behaviors (see Definition 2). Further, any choice model $(\boldsymbol{\sigma}^P, \boldsymbol{\lambda}^P)$ that contains only partially-ranked customer behaviors can be transformed into an equivalent choice model $(\boldsymbol{\sigma}^C, \boldsymbol{\lambda}^C)$ exclusively composed of fully-ranked customer behaviors.*

We refer to Appendix A.1.5 for a formal proof of Theorem 1, which transforms a partially-ranked sequence into a set of fully-ranked sequences which is factorially large in the size of the indifference set. This transformation only serves to prove equivalency, and one may find tighter upper bounds for the number of required fully-ranked sequences (see online supplement: Appendix B.1.1).

Theorem 1 suggests that fully- and partially-ranked choice models can essentially reflect the same set of customer behaviors, but the latter will tend to have a sparser representation. Choice models with full ranks can reflect preference indifference on products, but it will require several fully-ranked lists to represent an equivalent choice model. Further, explicitly ranking all products may be unnecessary, since low ranked products tend to have little impact in explaining sales, i.e., the proportion of the overall transaction predicted by them tends to be small. Quantifying the impact of a low ranked product may be difficult in practice, given that assortments are typically composed by strategically chosen items. However, for the purpose of illustration, we may quantify the impact of low ranked products on an *average assortment*, as stated below.

Observation 1 ***(Impact of low ranked products in explaining sales)****:* *The impact of low ranked products in explaining the sales transactions can be negligibly small, both statistically and in practice. Consider a non-empty assortment $\mathcal{S}$, let $r = \frac{|\mathcal{S}|}{N} \in]0, 1]$ be the assortment density of $\mathcal{S}$ and assume that the probability that a certain product is part of $\mathcal{S}$ is uniform (i.e., it is equal to $\frac{|\mathcal{S}|}{N}$). Then, the impact of a product decreases exponentially fast with its rank. Further, the greater the ratio r, the smaller the importance of low ranked products.*

A verification of Observation 1 can be found in Appendix A.1.2. With a ratio of $r = 0.1$, the probability that none of the strictly ranked products is part of the assortment is, on average, about 3.87% for rank $k = 10$, but only about 0.05% for $k = 50$. With higher ratios, lower ranks quickly become insignificant. For example, with $r = 0.5$, the probability for $k = 10$ is as low as 0.05%. While the above examples assume a uniform popularity in the customer priorities and in the assortments, in practice, it is likely that popular products are both highly ranked and are part of the assortment. We therefore argue that, in practice, the above stated probability tends to be a pessimistic (upper) bound on the importance of low ranked products. In other words, we expect that the probability that low ranked products are relevant decreases even more in practice.

We now introduce a special case of the simple partially-ranked choice model , where the indifference set contains all products that are not strictly ranked, i.e., $I(\sigma) = \mathcal{N} \cup \{0\} \backslash P(\sigma)$.

Definition 3 ***(Partially-ranked customer behavior with complementary indifference set):*** *A partially-ranked customer behavior σ with complementary indifference set is defined as a simple partially-ranked behavior (see Definition 2), and imposing that $I(\sigma) = \mathcal{N} \cup \{0\} \backslash P(\sigma)$.*

While it may be possible to use the more general partially-ranked choice model (see Appendix A.1.1) in the proposed methodological developments, we will restrict our attention from now on to the simplified case as by Definition 3. Even though using indifference sets that are composed of all non-ranked products may seem unrealistic in practice, several reasons favor their use. First, such models are theoretically as sound as fully-ranked choice models, since their behaviors can be transformed into fully-ranked ones (see Theorem 1). Second, they implicitly define the indifference sets and therefore avoid the associated risks of overfitting. Third, they provide a clean data structure that can more easily be explored in an estimation method. Finally, such indifference sets also allow us to develop an intuition of their contribution on explaining the overall sales. Their *explanatory power* is analytically computed in the following for the illustrative case of an *average assortment.*

Observation 2 ***(Explanatory power of the indifference set):*** *Consider a consumer behavior $\sigma_k = (P(\sigma_k), I(\sigma_k))$ with estimated market probability λ_k and an assortment $\mathcal{S}$ selected uniformly at random with assortment density r. The expected explanatory power of σ_k, i.e., its contribution to the explanation of the overall sales of product $i \notin P(\sigma_k)$ in $\mathcal{S}$ amounts to $(1-r)^{|P(\sigma_k)|} \cdot \frac{\lambda_k}{|I(\sigma_k)|}$.*

A verification of Observation 2 is given in Appendix A.1.3. As a consequence, one can easily verify the average explanatory impact on products that are not strictly ranked in a partially-ranked consumer behavior. For example, if $N = 100$, $|\mathcal{S}| = 50$ (thus, $r = 0.5$), 5 products are strictly ranked (i.e., $|P(\sigma_k)| = 5$, $|I(\sigma_k)| = N - |P(\sigma_k)| = 95$) and $\lambda_k = 0.05$, the behavior σ_k explains only 0.0016% ($= (1-0.5)^5 \cdot \frac{0.05}{95}$) of the sales of any product $i \in I(\sigma_k)$. The impact of using such an indifference set on explaining the overall sales, for this example, amounts to 3.12% ($= (1-0.5)^5$). Therefore, exposing the decision maker to such a concise list of 5 products is sufficient in practice, given that these 5 products explain 96.78% of sales caused by this consumer type.

5.3.2 Learning Consumer Preferences

Before we can optimize future assortments, we need to estimate the probability $\mathbb{P}(i|\mathcal{S})$ that a product i is sold given that a random customer is exposed to assortment $\mathcal{S}$. We may compute this probability, once our choice model $(\boldsymbol{\sigma}, \boldsymbol{\lambda})$ is estimated. Given that there is a factorial number of different customer behaviors, a major challenge is to identify the set $\boldsymbol{\sigma}$ that is relevant for explaining the sales, as well as their corresponding probabilities $\boldsymbol{\lambda}$. In this section, we focus on how to efficiently learn those parameters that are consistent with the observed sales.

In line with the works of Farias et al. [12] and Bertsimas and Mišic [74], we assume that historical data is available in aggregated form[1], including a total of M assortments, given by set $\mathcal{M} = \{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_M\}$, as well as sales transaction data for each of them. Those sales are given in a vector $\boldsymbol{v} \in \mathbb{R}^{(N+1)\cdot M}$ consisting of all pairs (i, m), with $i \in \mathcal{N} \cup \{0\},\ m \in \mathcal{M}$, which represent the customers that have chosen option i when being presented assortment S_m. Note that it is assumed that such sales data also includes the no-purchase option 0, e.g., if it is accurately collected by the store or estimated by the store manager.

Our proposed estimation method fits into the general column-generation approach from Van Ryzin and Vulcano [81]. Relevant customer behaviors are generated in the sub-problems, while the probabilities for a given set $\boldsymbol{\sigma}$ of customer behaviors are estimated in the master problem. In particular, the latter consists in estimating probabilities $\boldsymbol{\lambda}$ such that the choice model $(\boldsymbol{\sigma}, \boldsymbol{\lambda})$ is consistent with sales probabilities $\boldsymbol{v}$. To this end, it has become quite common (see, e.g., Farias et al. [12], Bertsimas and Mišic [74]) to embed the product choice of each customer type within a matrix $\boldsymbol{A}$, which has one column for each customer preference sequence and one row for each product and assortment. We may compute this matrix $\boldsymbol{A} \in \mathbb{R}^{((N+1)\cdot M)\times K}$, such that an entry $A^k_{(i,m)}$ is set to 1 if customer k would choose product i from assortment $\mathcal{S}_m \cup \{0\}$. As a consequence, $\forall(k, m) :\ \sum_i A^k_{(i,m)} = 1$. Note that this definition of the matrix $\boldsymbol{A}$ differs from the one used by Van Ryzin and Vulcano [81], who work on dis-aggregated data and therefore define one row per period (and at most one transaction per period).

The estimation of probabilities can be performed based on several criteria. Instead of estimating the choice model that results in the highest worst-case revenue [12], we follow two recent approaches that either maximize the likelihood probability [81], or minimize the estimation error [74]. While we will be exploring the two objectives in our computational experiments, we will here emphasize the model development for the latter: minimizing the ℓ_1 error between historical sales observations $\boldsymbol{v}$ and sales predictions $\boldsymbol{A\lambda} = \sum_k A^k_{(i,m)}\lambda_k$ (i.e., the probability that a random customer chooses option i from assortment S_m). We therefore define the training error as:

$$\epsilon^{Tr} = \sum_{(i,m)} |\boldsymbol{A\lambda} - \boldsymbol{v}|_{i,m}.$$

Consider a given set of potentially relevant customer behaviors $\{\sigma^1, \sigma^2, ..., \sigma^K\}$, one may use a simple linear program to find the corresponding probability distribution $(\lambda_1, \lambda_2, ..., \lambda_K)$ that

[1]The data is aggregated over time and customers, i.e., we assume that any information about the customers (id, features) and the exact purchase moment is either unavailable or ignored to estimate the models.

results in the smallest error as follows:

$$\min_{\boldsymbol{\lambda},\boldsymbol{\epsilon}^+,\boldsymbol{\epsilon}^-} \quad \mathbf{1}^T\boldsymbol{\epsilon}^+ + \mathbf{1}^T\boldsymbol{\epsilon}^- \tag{5.1a}$$

$$\text{s.t.} \quad \boldsymbol{A}\boldsymbol{\lambda} + \boldsymbol{\epsilon}^+ - \boldsymbol{\epsilon}^- = \boldsymbol{v} \tag{5.1b}$$

$$\mathbf{1}^T\boldsymbol{\lambda} = 1 \tag{5.1c}$$

$$\boldsymbol{\lambda}, \boldsymbol{\epsilon}^+, \boldsymbol{\epsilon}^- \geq 0. \tag{5.1d}$$

Setting the Choice Matrix A. Bertsimas and Mišic [74] propose to set the choice matrix $\boldsymbol{A}$ for fully-ranked customer behaviors such that $\boldsymbol{A}\boldsymbol{\lambda} = \boldsymbol{v}$ by using entry 1 if customer k would choose product i from $\mathcal{S}_m \cup 0$. This notion does not hold in the context of our more general representation which may include indifference sets. For reasons of simplicity and without loss of generality, let us consider the slightly simpler case $(P(\sigma), I(\sigma), 0)$ in which the customer behavior σ consists only of a single list $P(\sigma)$ of preferred and strictly ranked products, followed by an indifference set $I(\sigma)$. We assume that the indifference set $I(\sigma)$ (in the more general case, we have $I^1(\sigma), \ldots, I^q(\sigma)$) is externally given, for instance by a marketing department. Alternatively, it can be learned or estimated in a previous step by identifying products with similar characteristics. We may then define the ranking entries of a customer behavior σ as follows:

$$\sigma(i) = \begin{cases} \text{rank of preference of } i & \text{if } i \in P(\sigma) \\ |P(\sigma)| & \text{if } i \in I(\sigma) \\ +\infty & \text{otherwise.} \end{cases}$$

The preferred products are ranked from 0 to $|P(\sigma)|-1$, whereas all products in the indifference set have equivalent rank $|P(\sigma)|$. In the general case with several indifference sets, those ranks will be computed as the rank of the previous preferred product plus 1. The rank of a product that is neither in a $P(\sigma)$ nor in $I(\sigma)$ is set to $+\infty$.

Setting the choice matrix $\boldsymbol{A}$ for a partially-ranked choice model has to respect $\forall(k,m)$: $\sum_i A^k_{i,m} = 1$, which we can achieve by a uniform distribution on the indifference set:

$$A^k_{i,m} = \begin{cases} 1 & \text{if } i \in \mathcal{S}_m \text{ and } \forall j \in \mathcal{S}_m \backslash \{i\} : \sigma^k(i) < \sigma^k(j), \\ \dfrac{1}{|I(\sigma) \cap \mathcal{S}_m|} & \text{if } i \in \mathcal{S}_m : \text{ and } \forall j \in \mathcal{S}_m, \sigma^k(j) = |P(\sigma)| \text{ or } \sigma^k(j) = +\infty, \\ 0 & \text{otherwise.} \end{cases} \tag{5.2}$$

In words, customer k chooses item i from assortment S_m with a probability of 1, if i is available in m and is the most preferred among all available items. The customer chooses i

with probability $\frac{1}{|I(\sigma) \cap \mathcal{S}_m|}$, i.e., with uniform probability among all items that are both in the indifference set and in the assortment, if none of the strictly ranked items is available. All other items are not selected: either because they are not in the assortment, or because they are neither strictly ranked nor in the indifference set. Given the definitions above, once a subset $\boldsymbol{\sigma}$ of customer preferences (consistent with Definition 6) is identified, the corresponding choice matrix $\boldsymbol{A}$ can be efficiently computed.

Learning Consumer Behaviors based on a Preference Tree. As noted by Bertsimas and Mišic [74], the linear program (5.1) is not tractable, given the factorial number of customer behaviors here considered, causing $\boldsymbol{A}$ and $\boldsymbol{\lambda}$ to be exponentially large. The authors therefore propose a column generation algorithm, which initializes problem (5.1) as Master problem with a small subset of promising columns (i.e., preference sequences). The algorithm then iteratively identifies possibly relevant columns and adds them to the Master problem. Let $\boldsymbol{\alpha}$ and ν be the dual values of constraints (1b) and (1c) after solving problem (5.1). To find columns with minimal reduced cost, the authors further propose the following mixed-integer program:

$$\begin{aligned}
\min_{z,a} \quad & -\boldsymbol{\alpha}^T\boldsymbol{a} - \nu && && (5.3a)\\
\text{s.t.} \quad & a_{i,m} \leq z_{ij} && \forall m \in \{1,...,M\},\ i,j \in S_m \cup \{0\},\ i \neq j && (5.3b)\\
& z_{ij} + z_{ji} = 1 && \forall i,j \in \{0,1,...,N\},\ i \neq j && (5.3c)\\
& z_{ij} + z_{j\ell} - 1 \leq z_{i\ell} && \forall i,j,\ell \in \{0,1,...,N\}\ i \neq j,\ i \neq \ell,\ j \neq \ell && (5.3d)\\
& \boldsymbol{z} \in \{0,1\},\ \boldsymbol{a} \in \{0,1\}.
\end{aligned}$$

The MIP (5.3) minimizes the total reduced costs of the corresponding product sequence defined by the $\boldsymbol{z}$ variables. Constraints (5.3b) ensure that $a_{i,m}$ can take a positive value only if product i is preferred to all other products in assortment S_m. The set of constraints (5.3c) and (5.3d) represent non-reflexivity and transitivity, respectively, in order to establish a strict order among all products. Unfortunately, system (5.3) is costly to solve, and one needs to find alternatives for practical purposes. The authors therefore use a local-search heuristic to find new columns with reduced costs. Both the heuristic and the exact model (5.3) generate fully-ranked customer behaviors, which may not be necessary in practice (see Observation 1). We therefore propose to strictly rank only products with high ranks (i.e., those that are considered more preferred) and to explicitly take advantage of the structure of the proposed choice model.

The partially-ranked customer behaviors with indifference subsets can be efficiently represented by a preference tree, in which explicitly listed nodes refer to strictly ranked products. While the presented methods also apply to the general case of partially-ranked customer behaviors (see Definition 6 in Appendix A.1.1), we will focus, without loss of generality, on the simpler case of $\sigma = (P(\sigma), I(\sigma))$ as specified in Definition 3. Here, the indifference set contains all nodes that are not strictly ranked, i.e., $I(\sigma) = \mathcal{N} \backslash P(\sigma)$, and therefore does not need to be explicitly listed in the tree. Figure 5.1 illustrates a small example with 3 products and the no-purchase option 0. In this example, a total of $|K| = 8$ customer behaviors has been generated. For instance, customer behavior σ_7 refers to a customer that prioritizes product 2, if present; if not, she is willing to buy product 1. If none of those products is available, the customer will buy any available product or leave the store with equal (i.e., uniform) probability. In contrast, customer behavior σ_6 refers to a customer that will buy product 2, if available, and leave the store without purchase otherwise (indicated by the no-purchase option 0).

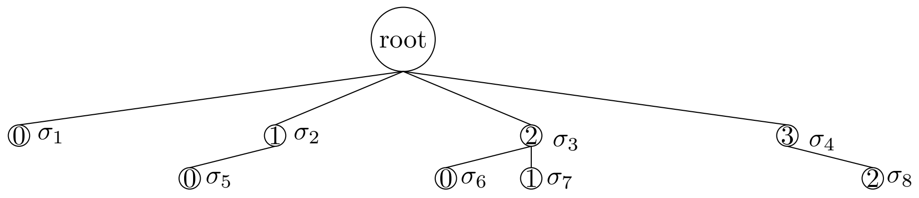


Figure 5.1 Example of Growing Preference Tree choice model for $N = 3$ products

Such a tree structure may be more intuitive for store managers who may want to understand customer segmentation, since it focuses on the products that are important to explain sales: those that are ranked early in a customer preference sequence. Further, the search for new customer behaviors in the tree structure may drastically speed up computation if one succeeds to limit the search to a significantly smaller space. This is the case if we focus on high ranks first and then gradually expand the tree by not more than one level of depth at each branch and iteration. Due to the gradual expansion of the tree, we will refer to it as the *Growing Preference Tree (GPT).*

We define σ_j to be a *sub-behavior* of σ_i if $P(\sigma_j) = (P(\sigma_i), \ell)$, where $\ell \in I(\sigma_i)$. In words, a sub-behavior σ_j inherits the strict preference list from its parent σ_i and adds to it one product (including, potentially, the no-purchase option) that is not part of σ_i's preference list. Let $\boldsymbol{\sigma}$ be the set of behaviors enumerated in the GPT. Our algorithm iteratively searches for new sub-behaviors in the GPT, expands the tree and solves problem (5.1) to find the corresponding probabilities $\boldsymbol{\lambda}$. When looking for new promising columns (i.e., new sub-behaviors), we may restrict the search to the sub-behaviors of all $\sigma \in \boldsymbol{\sigma}$. The reduced costs of each of the sub-behaviors can be computed as $rc(\sigma) = -\boldsymbol{\alpha a} - \nu$, where $\boldsymbol{\alpha}$ and ν are the dual values of constraints (1b) and (1c) in problem (5.1), and a is defined according to equality (5.2). The sub-behaviors with the lowest reduced cost are then added to the set of customer behaviors $\boldsymbol{\sigma}$, and problem (5.1) is resolved. The pricing step is exemplified in Figure 5.2, in which the reduced cost for all sub-behaviors (indicated in blue) of behaviors $\sigma \in K$ are computed, unless the last product in the preference list of σ is option 0. This would indicate that the customer would leave the store and the sub-behaviors are irrelevant. The process is performed iteratively until the ℓ_1 error is sufficiently small or a defined maximum number of iterations is performed.

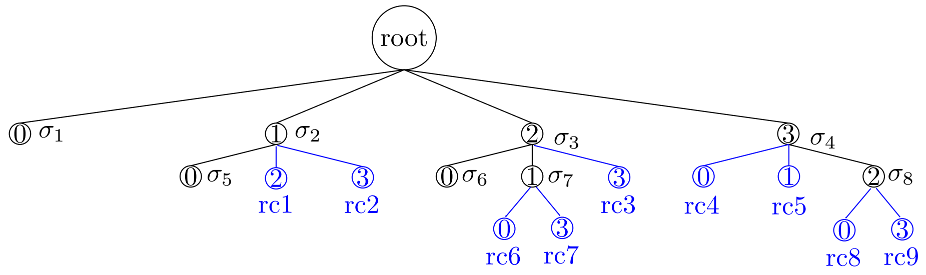


Figure 5.2 Computing reduced costs in the Growing Preference Tree choice model

Note that computing the reduced cost of a fully-ranked preference list has a computational complexity of $O(N \cdot M)$, even if the reduced cost of a "similar" fully-ranked preference list is known. Therefore, finding new columns with negative reduced costs can become costly when using fully-ranked preference lists. In contrast, using partially-ranked preference lists with the GPT holds the remarkable advantage that, once the reduced cost of a partially-ranked list is known, the reduced cost of any of its sub-behaviors can be computed in $O(M)$ time. The reduced cost of the sub-behavior (that will additionally rank item i) can be quickly adjusted by accounting for the different choices that may occur in each of the M assortments. It turns out that the reduced cost is impacted only in the case when the original preference sequence selected items from the indifference set, but the new sub-behavior would select item i, if $i \in S_m$. In this case, the corresponding dual value $\alpha_{i,m}$ has to be added, and one has to subtract the previous contribution of the indifference set from the reduced cost. This makes the exploration of the search space for negative reduced cost columns extremely efficient. A detailed description can be found in Appendix A.2.2.

The general steps of the GPT-based column generation method can be summarized as follows:

1. Generate initial behaviors and add them to the tree.

2. Compute reduced costs of sub-behaviors of all behaviors and add those with most negative reduced costs to the tree.
3. If no negative reduced cost column has been found, execute MIP (5.3) to find the most negative reduced cost column; if the cost is negative, add the behavior to the tree.
4. If none of the columns found in the previous two steps has negative reduced cost, terminate the algorithm. Otherwise, return to step 2.

A detailed description and pseudo-code of the algorithm can be found in Appendix A.2.1. Note that, in our computational experiments, we have never encountered cases where solving MIP (5.3) was necessary, given that the previous steps always found sufficient sub-behaviors with negative reduced cost to reach the required accuracy threshold. Contrary to directly using MIP (5.3) to identify the customer behavior with the lowest reduced cost, the GPT allows for controlling which type of customer behaviors to consider. For example, if indifferent sets are not at all desired by the modeler, one only needs to consider the sub-behaviors that end in the no-purchase option 0 and set matrix $\boldsymbol{A}$ accordingly. The GPT would then only generate strictly ranked customer behaviors, but most likely converge much faster than when using the MIP (5.3) or a local search.

We close this section by noting that, even though we have illustrated the model estimation by minimizing the linear ℓ_1 loss function, the general methodology of the GPT can be used with any convex loss function. We discuss those implementations in online supplement: Appendix B.1.6 by using the non-linear loss function of the Kullback Divergence [33].

5.4 Assortment optimization

In the previous sections, we have presented a new representation for rank-based choice models and an efficient methodology to identify a set $\boldsymbol{\sigma}$ of relevant customer behaviors $\sigma_k \in \boldsymbol{\sigma}$, as well as their corresponding probabilities λ^k. We now focus on how to identify optimal assortments that are coherent with the learned choice model.

Aouad et al. [88] recently showed that assortment optimization based on a given choice model is generally NP-hard. Some authors therefore proposed to limit the search space in order to remain computationally tractable. Jagabathula [89] proposes a local search for assortment optimization relying on a revenue prediction subroutine based on a general choice models. Honhon et al. [67] provide efficient algorithms for several special cases of rank-based models with $O(N^2)$ customer types, where N is the number of products. Berbeglia and Joret [47] provide tight bounds for the performance of so-called revenue-ordered assortments that respect the *regularity assumption*, therefore including the general class of Random Utility Models.

Restricting the number of products that each customer may want to buy to smaller *consideration sets* has also found more popularity. In this spirit, Aouad et al. [87] solve the assortment optimization problem with consideration sets via dynamic programming. Jagabathula and Rusmevichientong [90] jointly solve the price and assortment optimization problem under a two-stage choice process where customers first consider a subset of products cheaper than a certain price threshold and then choose their favourite product among them.

First general MIP formulations that consider the entire search space have been introduced by McBride and Zufryden [91] and Belloni et al. [55]. Initially, those models were limited to small problem instances. Nevertheless, some recent works have proposed MIP formulations that scale reasonably well (see, e.g., Farias et al. [12], Bertsimas and Mišic [74], Bertsimas and Mišić [56]). We will use those works as a starting point to optimize on partially-ranked choice models.

We suppose that a revenue r_i is associated with each product i. The no-purchase option 0 yields a revenue of 0. Bertsimas and Mišic [74], Bertsimas and Mišić [56] propose a mixed-integer programming model that uses variables x_i that take value 1 if product i is included in the assortment, and 0 otherwise. It further uses variables y_i^k that take value 1 if product i is within the assortment and is chosen according to behavior σ_k. The optimization problem writes as

$$
\begin{aligned}
\max_{x,y} \quad & \sum_{k=1}^{K}\sum_{i=1}^{N} r_i \lambda^k y_i^k && && \text{(5.4a)}\\
\text{s.t.} \quad & \sum_{i=0}^{N} y_i^k = 1 && \forall k \in \{1,...,K\} && \text{(5.4b)}\\
& y_i^k \leq x_i && \forall k \in \{1,...,K\},\ \forall i \in \{1,...,N\} && \text{(5.4c)}\\
& \sum_{j:\sigma^k(j)>\sigma^k(i)} y_j^k \leq 1 - x_i && \forall k \in \{1,...,K\},\ \forall i \in \{1,...,N\} && \text{(5.4d)}\\
& \sum_{j:\sigma^k(j)>\sigma^k(0)} y_j^k = 0 && \forall k \in \{1,...,K\} && \text{(5.4e)}\\
& \boldsymbol{x} \in \{0,1\},\ \boldsymbol{y} \geq 0.
\end{aligned}
$$

The optimization problem (5.4) maximizes the expected revenue. Constraints (5.4b) select exactly one product for each customer type k. Constraints (5.4c) say that a product can be selected only if it is part of the assortment. Constraints (5.4d) guarantee that the product in the assortment that is ranked highest also has the highest y_i^k value. Finally, constraints

(5.4e) ensure that all products ranked lower than the no-purchase option 0 are not selected.

For reasonably sized problem instances, Problem (5.4) is computationally tractable. In particular, even though defined as continuous variables, variables $\boldsymbol{y}$ will only take binary values due to the structure of the problem. Bertsimas and Mišić [56] explicitly showed that the formulation yields stronger linear programming relaxation bounds than the formulation initially proposed by Belloni et al. [55]. Unfortunately, the formulation requires fully-ranked customer behaviors, which are not explicitly given by a choice model with partially-ranked behaviors as generated by the GPT column generation algorithm. To adapt partially-ranked behaviors to the MIP stated above and obtain an exact approach, we could transform the partially-ranked choice model into a fully-ranked choice model. However, the number of necessary fully-ranked preference lists is factorially large (see Theorem 1), which would make the resulting MIP intractably complex. As an heuristic approximation, one may replace the indifference sets by a random sequence of those products that are not strictly ranked. Generating several random sub-behaviors for each partially-ranked customer's behavior, generally known as *boosting*, may be computationally tractable while improving the performance of the final assortments. However, it may be quite instance specific how many of those random sequences to generate, and the models may become too large anyhow, without mentioning the heuristic nature of the method.

We are therefore interested in finding an optimization model that directly operates on partially ordered ranks, i.e., those that use a strict ranking on a subset of products and indifference on the remaining products. To directly operate on customer behaviors with indifference sets, we may add to problem (5.4) the following constraints, which enforce that y variables for products i and j have equal values if both products are part of the assortment and have equivalent rank in behavior k. Namely,

$$z_{ij} = x_i \cdot x_j \quad \forall i, j \in \{1, \ldots, N\} : i > j \tag{5.5}$$

$$|y_i^k - y_j^k| \leq 1 - z_{ij} \quad \forall k \in \{1, ..., K\};\ \forall i, j \in \{1, \ldots, N\} : i > j \text{ and } \sigma^k(i) = \sigma^k(j). \tag{5.6}$$

The introduction of variables z_{ij} and the linearization of constraints (5.5) and (5.6) would significantly increase the model size and the difficulty of solving the problem. Fortunately, an equivalent model can be achieved by adequate transformation and substitution of the new variables.

Theorem 2 *The feasible set of the optimization model composed by (5.4a) - (5.4e) and (5.5) - (5.6) is equivalent to the feasible set of the optimization model composed by (5.4a) - (5.4e)* *and the constraints (5.7)-(5.8).*

$$y_i^k - y_j^k \leq 2 - x_i - x_j \quad \forall k \in \{1,...,K\};\ \forall i,j \in \{1,\ldots,N\} : i > j \textit{ and } \sigma^k(i) = \sigma^k(j) \tag{5.7}$$
$$-y_i^k + y_j^k \leq 2 - x_i - x_j \quad \forall k \in \{1,...,K\};\ \forall i,j \in \{1,\ldots,N\} : i > j \textit{ and } \sigma^k(i) = \sigma^k(j) \tag{5.8}$$

We refer to Appendix A.1.6 for a proof of Theorem 2. Given the above equivalence, one may directly optimize the assortment based on partially-ranked consumer behaviors without introducing new variables. While the structure of problem (5.4) forces the continuous $\boldsymbol{y}$ variables to take binary values, adding constraints (5.7) and (5.8) breaks this structural property, allowing the variables to take any continuous value between 0 and 1. As outlined above, these constraints have an intuitive interpretation. They force y_i^k and y_j^k to take the same value if both products i and j are part of the assortment and have equal rank in customer behavior k. In this case, if none of the higher ranked products is available in the assortment, each of y_i^k for the indifferent products will take value $\frac{1}{|I(\sigma) \cap \mathcal{S}|}$, where $\mathcal{S}$ is the assortment defined by variables $\boldsymbol{x}$. Even though there is a quadratic number of constraints, those are only on the size of the indifference sets. If the indifference sets are large, one may generate those constraints on the fly, adding only those constraints that are violated in the linear programming solution in each of the Branch-and-Bound nodes.

5.5 Computational Results

We now focus on empirical experiments performed with the proposed non-parametric choice model. In Section 5.5.1, we will evaluate the performance of the choice model and the assortment optimization algorithms on synthetic data. In particular, we compare our approach to two existing benchmarks in Section 5.5.1.1, evaluating how well these models perform in terms of scalability and ability of learning the choice model. Assortment optimization on the synthetic data is performed in Section 5.5.1.2. Finally, in Section 5.5.2, we will train the choice models on an industrial data sets from the clothes retail sector.

We note that additional analysis and results can be found in the appendices. In particular, online supplement: Appendix B.1 focuses on additional results for the estimation method, including, among others, the characteristics of the generated choice models when modifying parameters such as accuracy thresholds, ground truth model, or the loss function. Online supplement: Appendix B.2 focuses on additional results for the assortment optimization problem.

All computational experiments have been carried out on a single Intel Xeon X5650 2.67GHz processor, limited to 40 GByte of memory. The algorithms have been coded in Python version 3.5. If not stated otherwise, all mathematical models have been solved using the MIP solver of Gurobi version 8.0.1.

5.5.1 Numerical Results on Synthetic Data

Data Generation. We generate sales and assortment data according to a ground-truth (GT) model. This data will be used to evaluate the performance of the non-parametric choice models discussed so far and the corresponding algorithms proposed in the previous sections. We choose as GT model a Mixed Multinomial Logit model with T classes of customers. The probability distribution among the classes are drawn from the T-dimensional simplex p_t (therefore, $\sum_t p_t = 1$ and $p_t \geq 0\ \forall t$). Each customer class t associates an utility $u_{t,i}$ with a product i. The overall probability that a random customer chooses a product i is given by

$$\mathbb{P}(i|S) = \sum_{t=1}^{T} p_t \frac{e^{u_{t,i}}}{e^{u_{t,0}} + \sum_{j \in S} e^{u_{t,j}}}.$$

The utilities for each customer class are generated as proposed by Bertsimas and Mišic [74]. Specifically, we generate a matrix q of the same dimension as u uniformly distributed on $[0,1]$. If not stated otherwise, 4 of the $N+1$ products from $\mathcal{N} \cup \{0\}$ are randomly selected for each customer class t. Those products are assumed to have high utilities for customer class t, computed as $u_{t,i} = log(10 * q_{t,i})$. The utilities for the remaining $N-4$ products are set to $u_{t,i} = log(0.1 * q_{t,i})$. The training and test sets for the choice models consist each of M assortments. In all experiments, M has been set to 20 to reflect a context where the number of historical observations is limited. Assortment densities r are set to 0.5, i.e., each assortment contains $N/2$ products. Utilities $u_{t,i}$ are translated to a vector of sales probabilities $v_{i,m} = \mathbb{P}(i|S_m)$ for each product in each assortment. Sales transactions are then randomly generated according to the sales probabilities. Recall that the sales vector $\boldsymbol{v}$ is indexed by tuples (i,m). Therefore, the choice matrix $A^k_{i,m}$ has two dimensions.

Estimation of the Choice Models. The choice model is trained by iteratively generating new customer preference lists via column generation and solving the master problem which minimizes the estimation error based on the training data. Let $\boldsymbol{A}_{tr}$ and $\boldsymbol{v}_{tr}$ be the choice matrix and the sales probabilities for the M assortments of the training set. At each iteration, we compute the current training error as $\epsilon_{tr} = |\boldsymbol{A}_{tr}\boldsymbol{\lambda} - \boldsymbol{v}_{tr}|$. The test error is computed in the same way, but using choice matrix $\boldsymbol{A}_{te}$ and sales vector $\boldsymbol{v}_{te}$ based on the M assortments of the test data. Note that the scale of the training and test errors depends on the number of assortments in the training and test data. We may therefore normalize the errors, dividing by $2 \cdot M$ to obtain error values between 0 and 1. The estimation procedure then terminates once the normalized training error is smaller or equal to threshold ϵ_0 (i.e., we stop when $\epsilon_{tr} \leq 2 \cdot M \cdot \epsilon_0$). If not otherwise stated, we set ϵ_0 to 0.01.

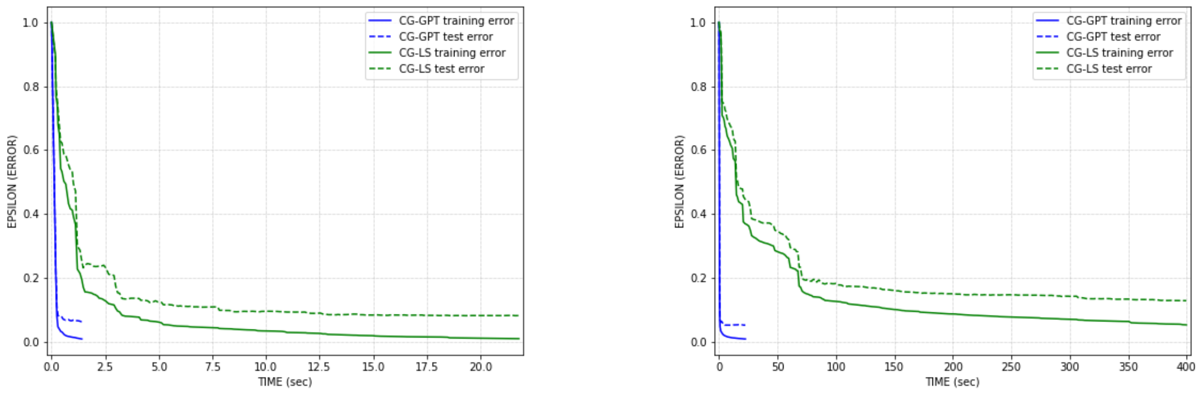
5.5.1.1 Estimation of Choice Models.

In the following, we will investigate how well the proposed choice model can be trained using our column generation approach based on the Growing Preference Tree (CG-GPT). We will compare our model with two existing approaches: the column generation approach from Bertsimas and Mišic [74], using a local search to find new fully-ranked preference lists (CG-LS), and the k-deletion heuristic from Jagabathula and Rusmevichientong [33]. All approaches have been executed on the same set of 10 randomly generated instances for each $N \in \{30, 50, 100, 250, 500, 1000\}$. Computing time has been limited to 12 hours.

Table 5.1 Learning performance for CG-GPT and CG-LS algorithms with $\epsilon_0 = 0.01$ (averaged over 10 random instances)

	CG-GPT					CG-LS					
N	Train error	Test error	# iter	time (min)	K	Train error	Test error	# iter	time (min)	K	# inst. unsolved
30	0.37	2.88	15.9	<1	170.7	0.39	3.74	309.0	1.5	177.5	0
50	0.38	2.90	31.1	<1	310.5	0.40	4.22	619.0	8.4	337.3	0
100	0.39	2.72	46.9	<1	619.0	0.40	4.41	1,183.1	76.3	667.7	0
250	0.39	2.37	90.6	16.6	1,474.1	-	-	-	-	-	10
500	0.40	1.89	116.5	76.1	2,437.1	-	-	-	-	-	10
1,000	0.40	1.30	159.7	306.0	3,876.7	-	-	-	-	-	10
all (avg)	0.39	2.34	76.8	66.6	1,481.4	0.40	4.13	703.7	28.7	394.2	30

Comparison with benchmark CG-LS. Table 5.1 reports the average results over the 10 instances for two approaches and for different problem sizes N. The table reports the final and test training errors, the computing times, the number of iterations, and the final number K of generated preference lists with non-negative probabilities λ_k. Column "# inst. unsolved" indicates the number of instances that have either hit the 40 Gbyte memory limit or run out of time. Those instances have not been considered in the average values. The CG-GPT has successfully trained the choice model to the required threshold of $\epsilon_0 = 0.01$ (which corresponds to a training error of 0.4 when $M = 20$) for all instances within the given time and memory limits. In contrast, the CG-LS is not able to converge for instances with more than 100 products. In a direct comparison, the CG-GPT is more scalable, converges much faster and provides better test errors than the CG-LS. In particular, one observes that the number of iterations required by the CG-GPT does not increase much as N increases. This is explained by the fact that the indifference sets have a larger explanatory power in those instances, which allows the model to have small training errors with relatively few columns. These are direct practical implications of Lemma 1 and Theorem 1. In fact, further experiments have shown that restricting the number of columns to those of highest probability has relatively little impact on CG-GPT test error, while CG-LS performance significantly deteriorates in this case. Detailed results and further discussions can be found in online supplement: Appendix B.1.7.



(a) Problem instance with $N = 30$ (b) Problem instance with $N = 100$

Figure 5.3 Learning curves (normalized training and test errors) for CG-GPT and CG-LS on two example problem instances.

Figure 5.3 (a) and (b) visualize the evolution of the normalized training and test errors for two problem instances with $N = 30$ and $N = 100$, respectively. We notice that CG-GPT quickly converges in both cases, achieving better test errors than CG-LS in much shorter time. Also, with 100 products, CG-LS was not able to converge within the reported time interval.

Note that, to make sure that we are comparing with a fair implementation of CG-LS, we have tried different neighborhoods explored by the local-search heuristic of CG-LS. However, those implementations did not improve its convergence and were computationally limited to $N < 250$. Detailed results are reported in online supplement: Appendix B.1.2.

Comparison with benchmark k-deletion heuristic. Based on the idea that one only needs to rank k items when at most $k-1$ items are missing in any assortment, Jagabathula and Rusmevichientong [33] propose to use the k-deletion heuristic which enumerates all preference sequences with k strictly ranked products. The computational burden of such an approach quickly increases when k becomes bigger. This technique was therefore originally proposed to deal with assortments where the number of missing products is much smaller than the total number of products. Given its computational burden, we test the k-deletion approach only for small values of k, namely 2, 3 and 4, on assortments of size $N/2 >> k$. Since in this case, none of the ranked products might be in the assortment, we add the no-purchase option 0 at rank $k+1$ (if not ranked before).

Table 5.2 Properties of choice models generated with the k-deletion method, with an assortment density $r = 0.5$

		k-deletion				
k	N	Train error	Test error	time (min)	K	# inst. unsolved
2	30	6.80	9.57	<1	93.7	0
2	50	7.03	10.65	<1	179.2	0
2	100	6.14	10.47	<1	400.4	0
2	250	5.66	10.32	<1	1,155.8	0
2	500	4.46	10.33	3.9	2,530.9	0
2	1000	3.73	10.40	12.3	5,499.8	0
2	all (avg)	5.64	10.29	2.8	1,643.3	0
3	30	1.90	5.69	<1	196.7	0
3	50	1.85	6.27	<1	362.7	0
3	100	1.04	6.38	6.4	857.5	0
3	250	0.56	6.16	120.1	2,351.8	4
3	all (avg)	1.34	6.12	31.8	942.2	24
4	30	0.02	3.98	2.3	278.4	0
4	50	0.03	4.36	20.2	479.4	0
4	100	0.00	5.46	493.0	980.0	9
4	all (avg)	0.02	4.60	171.8	579.3	39

Table 5.2 reports the computation results,[2] confirming that the k-deletion method does not scale well when increasing k. With $k = 3$, one is limited to 250 products, and with $k = 4$, one is limited to 100 products (note that we have omitted lines for N where none of the 10 problems were solved by k-deletion). The method is therefore not competitive with the CG-GPT, which achieves better training and test errors with a much smaller number of behaviors

[2]Instead of using the Gurobi solver, we here used CPLEX 12.7.1 with Python interface, since it turned out to build the model faster given the large number (sometimes millions) of columns.

(see, e.g., Table 5.1 above and Table 10 in online supplement: Appendix B.1.5). For small k, the k-deletion heuristic seems not flexible enough to provide a good fit of the data. For $k = 4$, the small training errors, but rather large test errors may indicate overfitting (see also online supplement: Appendix B.1.3). One possible cause may be the no-purchase option ranked after the first k strictly ranked products. On unseen assortments, the no-purchase probability is likely overestimated, particularly when the assortment density r is small. We here note that for CG-GPT, when using an accuracy threshold of $\epsilon_0 = 0.1$ such that it also strictly ranks not more than 3 products, the average test error is almost 40% smaller than the one of k-deletion with k=3 (see online supplement: Appendix B.1.5). This illustrates the importance of the complementary indifference set to improve predictive accuracy.

We close this section by referring the reader interested in further analysis and results of the proposed estimation procedure to online supplement: Appendix B.1. For example, our approach is not limited to linear loss functions, but can be used to minimize any other convex loss function (since strong duality still holds to prove optimality; see e.g., Griva et al. [92], Theorem 14.37). online supplement: Appendix B.1.6 numerically compares the CG-GPT with different objective functions, providing empirical evidence of the robustness and flexibility of the proposed technique. An analysis of choice model characteristics such as sparsity and concision can be found in online supplement: Appendices B.1.4 and B.1.5

5.5.1.2 Assortment Optimization.

In the previous section, we have shown that the partially-ranked choice model can be accurately learned in reasonable computing times even in contexts with large numbers of products. We now explore the scalability of the mathematical models to optimize assortments once the choice model has been learned. We have implemented four different approaches. The first two approaches learn a partially-ranked choice model using the CG-GPT algorithm. The first approach performs subsequent assortment optimization adding the indifference inequalities (5.7) and (5.8) via branch-and-cut (referred to as AO-B&C). The second approach performs boosting to create several fully-ranked preference lists at random and then optimizes via the classical MIP (5.4) based on fully-ranked lists (referred to as AO-Boost). Finally, the two other approaches are used as benchmarks, namely, one based on the CG-LS, and one based on the k-deletion heuristic. Both are used with the classical MIP (5.4) (referred to as AO-Compl). Given that AO-Boost is an approximation of AO-B&C and has been clearly outperformed by the latter, we here focus on the direct comparison with the two other approaches. The computational comparison with AO-Boost can be found in online supplement: Appendix B.2.1.

Comparison with CG-LS based approach. We now investigate how well the assortment optimization approaches for CG-GPT and CG-LS scale to a large number of products. For all experiments, we have used the choice model obtained after 12 hours of training with a 40GB memory limit, no matter whether or not estimation had converged by then. The second set of experiments assumes that the choice model is composed by the 100 customer preferences that have highest λ probabilities within the choice model.[3]

Table 5.3 Assortment optimization results for choice models generated by CG-GPT and CG-LS (averaged over 10 random instances) with $\epsilon_0 = 0.01$.

			CG-GPT - AO B&C				CG-LS - AO-Compl			
	N	Opt GT revenue	K	time (min)	GT revenue	Opt gap %	K	time (min)	GT revenue	Opt gap %
$\epsilon_0 = 0.01$	30	78.73	170.7	<1	78.05	0.00	177.5	<1	76.37	0.00
max K = full	50	84.18	310.5	<1	83.84	0.00	337.3	<1	81.76	0.00
	100	88.51	619.0	3.6	88.24	0.00	667.7	<1	87.31	0.00
	250	91.61	1474.1	83.4	91.39	0.00	1,175.0	16.4	90.65	0.00
	500	93.51	2437.1	557.5	93.38	0.30	413.7	13.7	90.92	0.00
	1,000	95.48	3876.7	720.2	93.00	5.04	150.8	10.2	92.75	0.00
	all (avg)	88.67	1,481.4	221.9	87.98	0.89	487.0	6.9	86.63	0.00
$\epsilon_0 = 0.01$	30	78.73	96.3	<1	78.05	0.00	99.2	<1	77.09	0.00
max K = 100	50	84.18	92.2	<1	83.85	0.00	98.7	<1	80.76	0.00
	100	88.51	90.2	<1	88.30	0.00	98	<1	86.14	0.00
	250	91.61	90.4	<1	91.39	0.00	99.3	<1	88.46	0.00
	500	93.51	89.5	1.4	93.40	0.00	100	2.2	87.84	0.00
	1,000	95.48	86.8	4.2	95.42	0.00	100	5.5	91.56	0.00
	all (avg)	88.67	90.9	1.1	88.40	0.00	99.2	1.4	81.33	0.00

Table 5.3 summarizes the results and reports for each problem size N the averages of the optimal ground-truth revenues (computed, as proposed by Bront et al. [93]). For each approach (i.e., CG-GPT and CG-LS), the table reports the average size K of the choice models, the average computing times to solve the optimization model, the average revenue as computed by the ground-truth model, and the average optimality gap after 12 hours of assortment optimization.

In the first set of experiments using the entire choice models trained, K inevitably grows as the number of products N increases. It must be noted that the CG-LS would generally also have growing K. However, for large N the choice models could not be solved within the time limit (given that the local search is more time-consuming; see Table 5.1), which results in small choice models (and high test errors). The AO-B&C approach based on the CG-GPT consistently generates assortments with higher GT revenues. However, with

[3]To be precise, after selecting the 100 preferences sequences with highest probabilities λ, the restricted Master problem has been re-solved for this subset to estimate the new probabilities.

larger N it quickly becomes more difficult to solve, as the number of preference sequences K becomes prohibitively large, requiring to add too many inequalities (5.7) and (5.8) when using indifference sets.

However, as previously discussed, partially-ranked choice models tend to achieve better generalization with a sparser model than fully-ranked ones (we refer the reader again to Section 5.3.1 and to online supplement: Appendix B.1.7). Furthermore, using a smaller set of preference sequences directly facilitates the solution of the assortment optimization problem. The second set of experiments, limiting the number of preference sequences to 100, yields convincing results. All models are solved to optimality within a few minutes. The CG-LS based approach tends to significantly loose prediction accuracy, resulting in assortments of lower quality (i.e., lower GT revenue). In contrast, the CG-GPT based approach maintains its predictive performance and results in assortments that have a close-to-optimal GT revenue. We conclude that limiting the number of preference sequences used in the assortment optimization seems to be a promising technique to remain computationally tractable while maintaining a choice model with high predictive performance. We refer the interested reader to online supplement: Appendix B.2.3 for the results of further computational experiments in which the number of preference sequences has been limited.

Comparison with k-deletion based approach. As a last study, we investigate the performance of assortment optimization based on the choice models obtained by the k-deletion heuristic. Table 5.4 compares the revenues of the assortments obtained by CG-GPT with AO-B&C, and the k-deletion with AO-Compl. Column "# inst unsolved" reports the number of instances that could either not be estimated by the k-deletion heuristic or not be solved in the assortment optimization model (exceeding the available memory). Lines without any results (i.e., the "# inst unsolved" equals 10) have been omitted. The results, averaged only over instances where the k-deletion heuristic was tractable, confirm that the k-deletion is not an appropriate choice in our context. For the choice model estimation, one can only enumerate sequences with up to 4 items, which is not sufficient to generalize well, resulting in assortments with lower GT revenues. As N gets larger, the choice model become too large, typically hitting the memory limit. Except for one exception ($N = 100$ with $k = 4$), the GT revenues reported by the k-deletion heuristic were consistently inferior to those of the CG-GPT based approach.

We refer to online supplement: Appendix B.2 for further experiments on the performance of the assortment optimization. In particular, online supplement: Appendix B.2.2 investigates the revenue impact of different loss functions and training accuracy thresholds. In online supplement: Appendix B.2.4 we tested the scalability of MIP (5.4) on a set of hard instances

Table 5.4 Assortment optimization with k-deletion compared against optimal revenues and revenues obtained by CG-GPT on those instances solved by both methods.

				k-deletion - AO-Compl			
k	N	Optimal GT revenue	CG-GPT GT revenue	K	time (min)	GT revenue	# inst unsolved
2	30	78.73	78.05	93.7	<1	74.78	0
2	50	84.18	83.84	179.2	<1	80.11	0
2	100	88.51	88.24	400.4	<1	85.76	0
2	250	91.61	91.39	1,155.8	29.3	85.52	0
2	500	93.51	93.38	2,530.9	459.3	88.22	0
2	all (avg)	87.31	86.98	872.0	97.8	82.88	10
3	30	78.73	78.05	196.7	<1	69.10	0
3	50	84.18	83.84	362.7	<1	79.37	0
3	100	88.51	88.24	857.5	1.8	85.08	0
3	250	93.07	92.82	2,351.8	240.5	84.18	4
3	all (avg)	86.12	85.74	942.2	60.6	79.43	24
4	30	78.73	78.05	278.4	<1	74.27	0
4	50	84.18	83.84	479.4	<1	81.74	0
4	100	96.65	95.88	980.0	2.2	96.17	9
4	all (avg)	86.52	85.92	579.3	<1	84.06	39

constructed following the results of Aouad et al. [88]. We conclude this section noting that the recent work of Bertsimas and Mišić [56] also conducted numerical experiments on the original formulation (5.4) for fully-ranked lists. The authors further propose a Benders decomposition implementation, which could also be adapted to our formulation. It has to be noted, however, that the assortment optimization MIP itself is not the only crucial ingredient of the overall approach. Learning the choice model correctly and accurately for large-scale problems is, as has been shown, computationally challenging and crucial to obtain meaningful assortment optimization models and, in this concern, we believe that the partially-ranked choice model provides an efficient option.

5.5.2 Case Study on Industrial Retail Data

We will now discuss an industrial case study based on real world data from a North-american clothes retailer. An anonymized data set on shoe stores has been obtained from our industrial partner JDA Labs [75]. In the following, we will discuss the data sets and preprocessing. We will then explore how well the different approaches perform when training the choice models for the industrial data set.

5.5.2.1 Data description and preprocessing

The data set includes assortment data, transaction data , as well as product and store characteristics from August 2014 to July 2015. Assortment information is provided for each day and each store (with information such as location, climate, price category). Sales transaction data contains product IDs, sold quantities, sales time-stamps and store ID. Products in the shoe dataset have characteristics given as categorical values (class, sub-class, brand, material, color) and continuous values (average price). Products in the shirt data set contain additional information (lifestyle, pattern, fit, sleeve length, fashion).

Based on those information, we define an assortment as the set of products offered in a particular store throughout one calendar week in a particular year. For each assortment, we link the corresponding sales transactions and convert those into the vector of sales probabilities $\boldsymbol{v}$, representing the probability of selling a certain product in a given assortment. The following describes the entire process of data preparation. For more details, we refer to the Master thesis of Palmer [94].

Store clustering. Assuming that stores in neighborhoods with similar characteristics meet the needs of similar customer types, we need to learn separately for groups of similar stores. As a result of the frequent discussions with JDA Labs [75], which provided the data, we clustered the stores according to four store features: location (state and city), climate (4 different categorical values), price band (low, medium and high), and percentage of sales in each sub-category. Given that our features contain both categorical and continuous values, the clustering has been performed using an extension of k-means that can handle mixed categorical and continuous data [95].

Data preprocessing and no-purchase estimation. We arbitrarily selected a cluster from the shoes data set that contains 10 stores and has a fairly high number of sales. For each of these stores, we use data during 10 consecutive weeks from Autumn 2014, which we consider a good trade-off to have sufficient data, while not risking that store assortments changed much due to seasonal fashion. We considered each week of store data as a proper assortment, resulting in a total of 100 assortments. To ensure that the historical data is statistically meaningful, we only considered products that have been sold at least 10 times, resulting in a final total of 299 different products.

Given that we did not have any information about how many customers left the store without purchase, we estimated the no-purchase probabilities assuming that assortments with many sales have lower no-purchase probabilities and assortments with few sales have higher no-purchase probabilities (but always between 10% and 30%). Let $v_{0,m}$ be the no-purchase probability in assortment S_m. Let $s^{\#}_m$ be the number of sales observed in assortment S_m. Let $s^{\#}_{MIN} = \min_{m \in \mathcal{M}}\{s^{\#}_m\}$ and $s^{\#}_{MAX} = \max_{m \in \mathcal{M}}\{s^{\#}_m\}$. The no-purchase probability for a store m is computed as a linear interpolation between 0.1 and 0.3 according to the number of observed sales: $v_{0,m} = 0.1 + 0.2 \cdot \frac{s^{\#}_m}{s^{\#}_{MAX} - s^{\#}_{MIN}}$. Let $s^{\#}_{i,m}$ denote the number of sales of product i in assortment S_m. We then compute the sales probability for any other product i as $v_{i,m} = (1 - v_{0,m}) \cdot \frac{s^{\#}_{i,m}}{s^{\#}_m}$, which is the corresponding sales proportion of product i taking into consideration the probability for the no-purchase option.

5.5.2.2 Computational Results

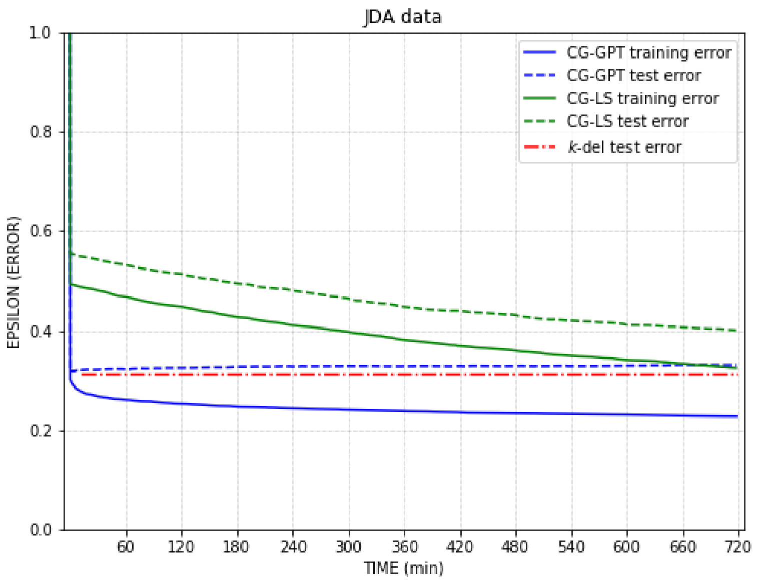


Figure 5.4 Learning curves (normalized training and test errors) for CG-GPT and CG-LS on industrial shoe data with 299 products.

Convergence of training the choice model. Figure 5.4 plots the convergence curves for the CG-GPT and CG-LS training approaches. To warm-start the CG-LS method, we initialized it with singleton preference lists as in Van Ryzin and Vulcano [81], i.e., we generate one preference list for each product, with that product being ranked first, and the no-purchase option ranked second. The initial set of preference sequences therefore corresponds to an independent demand model. We also plot the test error achieved by the k-deletion approach once the corresponding LP has been solved. Only the case for $k = 2$ has been reported, since higher values of k ran into memory errors.

We notice that the optimal training error is not 0 (as it is the case for synthetic data), because real data is noisy, or even contradictory.[4] Of course, one may group items in a pre-processing step as proposed by Jagabathula and Rusmevichientong [33], which would reduce data sparsity and improve both the training and the test errors. However, we refrain from using such techniques here due to two reasons. First, we did not find any grouping criteria which would be meaningful for the given context. Second, the primary interest of our study is to explore the efficiency of the methods when predicting sales probabilities for individual items.

The training error for the CG-GPT reaches about 25% in one hour, and about 22% after 12 hours. The CG-LS only reaches a training error of 32% after 12 hours. Test errors are, as expected, slightly higher than the training errors. For the CG-LS, the test error starts high and monotonically decreases to about 40% after 12 hours. For the CG-GPT, the test error finds its minimum (at about 31%) suprisingly fast, after a few iterations, and then slightly starts to overfit, increasing to about 33% after 12 hours. Remarkably, our method reaches that points after a few minutes, while the CG-LS never reaches it within 12 hours. Note that both methods start with a similar set of initial preference sequences. However, instead of ranking the no-purchase option 0 after the first product, the CG-LS uses the complementary indifference set. The initial training and test errors therefore confirm, once again, the importance of the indifference to explain the training data and to improve prediction accuracy on the test data. Interestingly, the k-deletion method achieves about the same level of generalization as the CG-GPT approach (with a test error of about 31%), requiring a similar amount of computing time (12 minutes for k-deletion and 6 minutes for the CG-GPT). This is a surprisingly good performance, since the k-deletion heuristic did not provide satisfactory results on synthetic data in previous experiments. We may conclude that for this particular industrial data set (with the techniques used to cluster the assortments), a more general partially-ranked choice model generalizes slightly better than a more refined one. On the other hand, capturing substitution of at least 2 dimensions seems to work well, since the CG-LS initialized with independent demand preference sequences (which is equivalent to a k-deletion with $k = 1$) did not yield good test errors, but could further be improved (gradually, by the the CG-LS itself, and by the k-deletion heuristic with $k = 2$). This, together with the fact that the training error generally remained quite high, might be indicators that the performance of rank-based models on certain industrial data sets may still be improved in future. Finally, also note that the same pattern has been confirmed by

[4]As an example, consider two assortments $S_1 = \{1, 2, 3\}$ and $S_2 = \{1, 2, 4\}$. We may have observed only sales of product 1 in assortment S_1, and only sales of product 2 in assortment S_2. In this case, a ranking-based choice model cannot perfectly fit the sales transactions for both assortments.

a 5-fold cross-validation analysis. After splitting the 100 assortments into 5 groups of 20 assortments each, 5 different experiments were carried using one of these groups as test set and the other 4 groups as training set. All experiments yielded similar conclusions.

5.6 Conclusion

In this work, we have focused on non-parametric rank-based choice models for assortment optimization. Those choice models have several advantages. Mainly, they can be estimated in a purely data-driven manner without relying on previous knowledge on the market structure. They also tend to be less sensitive to overfitting. Our work proposes a new methodology to estimate those choice models, which scales to a large number of products. In particular, we propose to represent customer behaviors not by fully ordered preference lists of all products, but only a subset of them. This is a realistic setting, which directly exploits the fact that products with low ranks have little explanatory power and impact in the buying behavior. Further, we show that any partially-ranked choice model can be transformed into an equivalent fully-ranked choice model, and vice-versa. The partial representation of the strictly ranked products enables us to efficiently train the choice model by gradually expanding a tree, in which each of the nodes represents partial lists of strictly ranked products. On this particular structure, new preference lists can be found efficiently via column generation. We finally present new inequalities to adapt an existing assortment optimization model to our partially-ranked choice models. Extensive computational experiments have shown that instances with up to 1,000 products can be efficiently trained and assortments can be optimized in quite low computing times, therefore significantly increasing the capabilities of previous approaches to learn the choice model. Our numerical experiments also revealed that the indifference sets are both useful to explain sales in the training data and to improve predictive accuracy on the (unseen) test data. Given that training the partially-ranked choice model by means of a growing tree and column generation has been proven to be very efficient in the case of assortment optimization, it may be a promising avenue to explore it any other context in which discrete choice models are central.

Acknowledgments The authors are thankful to JDA Labs, in particular to Marie-Claude Côté, for providing the industrial data sets and for their support throughout this research, and to the anonymous referees for their meaningful comments and suggestions. The work of the first author was supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada under grant 2017-05224. The third and the fourth author were, respectively, Research Master and Ph.D. students at Polytechnique Montréal in the Canada

Excellence Research Chair "Data Science for Real-time Decision-making". The fourth author has also been partially funded by MITACS, whose contribution is gratefully acknowledged.

CHAPTER 6 ARTICLE 3 - ON THE ESTIMATION OF DISCRETE CHOICE MODELS TO CAPTURE IRRATIONAL CUSTOMER BEHAVIORS

Authors: Sanjay Dominik Jena, Andrea Lodi, Claudio Sole
Under review in *INFORMS Journal on Computing*

Abstract The *Random Utility Maximization* model is by far the most adopted framework to estimate consumer choice behavior. However, behavioral economics has provided strong empirical evidence of *irrational* choice behavior, such as *halo* effects, that are incompatible with this framework. Models belonging to the Random Utility Maximization family may therefore not accurately capture such irrational behavior. Hence, more general choice models, overcoming such limitations, have been proposed. However, the flexibility of such models comes at the price of increased risk of overfitting. As such, estimating such models remains a challenge. In this work, we propose an estimation method for the recently proposed Generalized Stochastic Preference choice model, which subsumes the family of Random Utility Maximization models and is capable of capturing halo effects. Specifically, we show how to use partially-ranked preferences to efficiently model rational and irrational customer types from transaction data. Our estimation procedure is based on column generation, where relevant customer types are efficiently extracted by expanding a tree-like data structure containing the customer behaviors. Further, we propose a new dominance rule among customer types whose effect is to prioritize low orders of interactions among products. An extensive set of experiments assesses the predictive accuracy of the proposed approach. Our results show that accounting for irrational preferences can boost predictive accuracy by 12.5% on average, when tested on a real-world dataset from a large chain of grocery and drug stores.

6.1 Introduction

Accurately forecasting the demand of certain products or services is of crucial importance in the context of supply-chain optimization and retail operations. Most often, a predictive model must be learned from historical data representing the choice behavior of an agent faced with a discrete set of alternatives, called the *offer set*. A common assumption when dealing with demand estimation is to consider product demands as independent from each other, resulting in the independent demand model (see, e.g., Strauss et al. [18], Talluri and Van Ryzin [96]). However, it is well known that this assumption does not hold in many real-life scenarios and that product demands interact through substitution and halo effects. In general, we consider alternative x a substitute of alternative y if the presence of x in the offer set decreases the probability of y being chosen. On the contrary, we refer to an halo effect if the presence of x in the offer set increases the attractiveness of y, and thus its likelihood of being chosen.

Discrete choice models have been widely adopted to model substitution. Among them, the family of choice models that received the most attention in the literature is undoubtedly the one of *Random Utility Maximization* (RUM) models [97, 98, 6]. Choice models belonging to the RUM family assume that a random utility is assigned to every alternative. Utilities are modeled as random variables, and different choices about their distribution lead to different choice models. When faced with an offer set, the decision maker samples a vector of utilities and picks the option with the highest one, so as to maximize her expected payoff.

The standard theory of rational choice assumes the relative preference between two alternatives does not depend on the other products in the offer set. Hence, if alternative x is preferred to alternative y in a given offer set S, the same should hold for any other offer set $S' \neq S$. Starting from this assumption, known as *Independence of Irrelevant Alternatives* (IIA), Luce [6] derived the Multinomial Logit (MNL) model, which is arguably the most famous RUM choice model. Its popularity stems from the facts that it can be efficiently estimated, it is interpretable and, when used for decision making, it allows to benefit from appealing theoretical and computational properties. The Multinomial Logit model lacks, however, flexibility, imposing specific patterns of substitutions among alternatives. In particular, the logit formula implies that the ratio between the choice probabilities of two alternatives does not depend on the other (irrelevant) alternatives in the offer set. In response, many models have been proposed to overcome such limitations, so as to capture more complex patterns of substitutions. In the Nested Logit model, for example, this is achieved by assuming IIA holds only among groups of similar alternatives. In the Mixed Multinomial logit (MMNL) and Rank-Based [12] choice models, instead, violations of the IIA assumption are captured by aggregating several IIA-consistent classes of customers. Notably, some of these models, such as the MMNL, the Rank-Based and the Markov Chain [29] choice models, can theoretically approximate any RUM choice model and therefore capture arbitrarily complex substitution effects. We refer the interested reader to the computational study of Berbeglia et al. [32] for a deeper overview on RUM choice models and their generalization performances.

All models that belong to the RUM family obey the so-called *Regularity* assumption, which states that the introduction of an option in the offer set cannot increase the probability of another alternative being chosen. Hence, they cannot capture halo effects. Nevertheless, many studies in the literature of behavioral economics corroborated the reproducibility and robustness of this type of choice behaviors (see, e.g., Simonson [99], Huber et al. [100]), incompatible with the theory of utility maximization and therefore referred to as *irrational.* In the remainder of this paper, we therefore refer to rational behavior as one that can be captured by RUM models, and to irrational behavior as one that cannot. In order to better illustrate the kind of choice scenario in which violations of the Regularity assumption may arise, we report below the results of a choice experiment from the seminal work of Simonson and Tversky [13].

Example 6.1.1 (Choice Experiment from Simonson and Tversky [13]) *In this experiment, respondents were asked to choose among three camera models, which differ in terms of price and quality. Specifically, the three models were (1) a Minolta X-370 camera, priced at $170, (2) a Minolta 3000i, priced at $240 and (3) a Minolta 7000i, with a price of $470. Table 6.1 reports the market shares of the alternatives in the choice scenarios S_1, where only options {1,2} are offered, and S_2, where option (3) is added to the offer set. This experiment exhibits a violation of the regularity assumption, since the probability of choosing option (2) increases from 50% to 57% when option (3) is added to the offer set. Hence, no model belonging to the RUM class can perfectly fit this dataset.*

Table 6.1 Market share of three camera models in choice scenarios S_1, where respondents must choose between alternatives $\{1, 2\}$, and S_2, where option (3) is added to the offer set

		Market share	
Model	Price ($)	S_1	S_2
(1) Minolta X-370	170	.50	.22
(2) Minolta 3000i	240	.50	.57
(3) Minolta 7000i	470	-	.21

The choice phenomena reported in Table 6.1 is an example of the so-called *compromise effect*, where middle (i.e., compromise) options in terms of price and quality are preferred to extreme ones. One may be tempted to handcraft an utility function based on price and quality features in order to explain the observed choice outcomes. However, this cannot be done without considering the *assortment-dependent* effects on the attractiveness of products, which is inconsistent with the theory of rational choice.

Violations of the regularity assumption may be induced by other cognitive biases as well. For example, in the context of grocery shopping, when two complementary products (e.g., pasta and tomato sauce) are present in the assortment, the perceived attractiveness of both is likely to increase. One may also observe asymmetric, or *decoy* effects (see, e.g., Huber and Puto [101]) when the addition of an option (the decoy) to the offer set increases the choice probability of another alternative perceived as better. This choice phenomena was popularized a choice experiment reported in Ariely [102], in which a group of students was asked to choose among three possible subscription plans to "The Economist" magazine. For the sake of brevity, we report this experiment in Appendix C.1, where we also show how the GSP model (see Section 6.3) can explain the related choice outcomes. We refer the interested reader to Berbeglia [2] for more examples on the topic.

Such observations motivated a recent interest in more general choice models, capable of overcoming the limitations of the RUM framework. Unfortunately, many of these choice models lack efficient estimation schemes, and their performance on non-RUM instances has not been well understood yet (see, e.g., Jagabathula and Rusmevichientong [33]). Also, the minimal assumptions these models make about the distribution of choice probabilities may increase the risk of capturing spurious patterns from data, i.e., overfit.[1] This may be observed in the form of a model experiencing high variance in predictive accuracy when estimated on little amount of training data. Finding the delicate balance between flexibility and predictive accuracy is therefore of crucial importance for the practical utility of such models. The *Generalized Stochastic Preference* (GSP) choice model, an extension of rank-based choice models introduced by Berbeglia [2] to capture halo effects, is one of the recently proposed models that fits into this stream of literature. Despite being theoretically attractive, the estimation of the GSP choice model poses significant challenges both from the computational and predictive points of view. The authors suggest that estimation procedures originally developed for rational rank-based choice models (see, e.g., Farias et al. [12], van Ryzin and Vulcano [30], Bertsimas and Mišic [31]) may be adapted to their irrational choice model. Nevertheless, no empirical study has been reported in order to assess the estimation efficiency and predictive accuracy of the GSP choice model.

Next to computational challenges to estimate such model, its flexibility also comes at the price of an increased risk of overfitting. Even in the case of rational behaviors, it is known (see, e.g., Berbeglia et al. [32], Jena et al. [73]) that the estimation of general RUM models on little amounts of transactions data tends to be prone to overfitting. In the same line, on real-world data, Maragheh et al. [15] came to a similar conclusion. When estimated on small amounts of training data, their model, extending the MNL model to allow for pairwise

[1]Mostly a concern in Statistics and Machine Learning, *overfitting* refers to the situation when a model is too tailored to a specific data set (typically, the training data), and as such fails to generalize well to other data sets (e.g., the test data). Overfitting typically occurs either when the model is too general, or when the training data is not sufficiently representative for the ground truth. As a consequence, an overfit model may not yield accurate predictions on other data sets.

interactions among products, did not outperform a simpler MNL. Hence, particular care should be taken to such issue for effectively estimating the GSP choice model, which can account for even higher orders of interactions among products.

Contributions. In this work, we propose an estimation method for the GSP choice model. Specifically, we show how to use partially-ranked preferences to model irrational customer behaviors, and how to efficiently estimate them from choice data by adapting the column generation approach proposed by Jena et al. [73]. Partially-ranked preferences allow us to circumvent several difficulties regarding the adaption of estimation methods for strictly ranked preferences. In particular, our objective is to train the choice model so as to maximize its predictive accuracy. This is different from Farias et al. [12], who focus on worst-case revenue prediction for a given assortment of items. Also, our estimation method can easily handle both rational and irrational customer behaviors. In contrast, it is not clear how the Mixed Integer Programming (MIP) formulation of the Market Discovery subproblem from van Ryzin and Vulcano [30] should be adapted to allow for the discovery of irrational preferences. Finally, the *Growing Preference Tree* (GPT) algorithm of Jena et al. [73] provides a strong computational advantage in terms of scalability, especially important when dealing with irrational customer behaviors (discussed in the following) and generalizes well to unseen offer sets when tested on RUM instances. The application of partially-ranked preferences for tackling the estimation of generalized stochastic preferences thus looks promising. An appealing property of our approach stems from the fact that the irrationality, and thus the flexibility of the choice model is increased in an adaptive, data-driven way. By increasing the set of possible customer behaviors only when required to better explain the given data, we may limit the risk of overfitting and speed up the estimation procedure. To further reduce the risk of capturing spurious, high order interactions among products, we propose a new dominance rule among entering columns, prioritizing customer types with a small number of strictly ranked products and large indifference sets.

We run an extensive set of experiments to assess the predictive performance of the proposed choice model. Using the methodology delineated by Jagabathula and Rusmevichientong [33], we characterize the *rationality loss* of both generated and real instances. This allows us to observe that irrational customer types can significantly improve predictive accuracy on instances presenting halo effects among alternatives. We further show that our new criteria for discovering customer types can provide a further boost in predictive accuracy. Notably, our algorithm outperforms, on average, two baselines from the literature on irrational choice modeling, the Halo-MNL [15] and the Pairwise Choice Markov Chain (PCMC) [36], when tested on real-world data.

Organization of the paper. In Section 6.2, we review the literature on irrational choice models. In Section 6.3, we introduce the GSP choice model from Berbeglia [2] and our corresponding partially-ranked representation. We show how to estimate the proposed choice model in Section 6.4. The numerical results of our experiments on both synthetic and real instances are reported in Section 6.5. Finally, concluding remarks are reported in Section 6.6.

6.2 Related work

Our work spans several areas of research. In the following, we first review the literature from Psychology and Marketing, where several descriptive models, with little applicability from the predictive point of view, have been proposed to overcome the limitations of the RUM framework. We then survey the works from the Machine Learning and Operation Management communities, where discrete choice models of various levels of generality have been proposed.

Descriptive theories of choice. In order to define the notion of a *rational* agent, most economists rely on a set of consistency principles of rationality, which includes, among others, the aforementioned Regularity assumption and the more famous axiom of Independence of Irrelavant alternatives (IIA). This set of assumptions aims at describing how a rational agent is supposed to make her decisions across different offer sets. However, a vast body of literature has provided strong empirical evidence of choice behaviors incompatible with the theory of rational choice (we refer to Rieskamp et al. [103] for an excellent overview on the topic). The RUM framework is flexible enough to explain most of these choice behaviors, but cannot account for violations of the Regularity assumption. To overcome such limitation, more general theories of choices have been developed in psychology, such as Decision Field theory [104, 105] and the Leaky competing accumulator model [106]. These models belong to the broader class of Sequential Sampling models, which mimic the evolution of the decision-making process over time, and can account for violations of the rationality principles, including the Regularity one. They lack, however, practical estimation algorithms, and are usually adopted from a descriptive point of view more than a predictive one. Other works, such as Tversky and Simonson [107] and Rooderkerk et al. [16], embed alternatives into an attribute space, where context-dependent features are computed in order to determine the utility of each of the alternatives. These approaches have usually been applied to small, controlled experiments, and rely on the existence of two metric features, along which customer preferences are supposed to monotonically increase or decrease. This is a key difference with respect to our approach, where no item feature is supposed to be given.

Discrete choice models escaping RUM. Decomposing the utility into two components, item-specific and context-dependent, is also the starting point of Maragheh et al. [15] and Seshadri et al. [34], who propose a second-order extension of the MNL model in order to capture pairwise product interactions. However, these models do not subsume the RUM framework and thus, as pointed out by Jagabathula and Rusmevichientong [33], are not guaranteed to provide a better fit than RUM methods, even when applied to irrational instances. The same limitation holds for other models such as the Generalized Attraction Model from Gallego et al. [108] (see also [33]), the Perception-adjusted choice model [109] and the General Luce Model [110]. Feng et al. [111] propose a welfare-based framework, which subsumes the RUM framework and can be used to obtain choice models able to capture violations of the regularity assumption. The estimation of these choice models, however, is left by the authors as an open research question. Another general approach for which no empirical result has been reported is the Generalized Stochastic Preference choice model [2], an extension of rank-based choice models (see, e.g., Farias et al. [12], van Ryzin and Vulcano [30]) that allows for irrational customer behaviors. This model subsumes the RUM family of models and generalizes the non-RUM approach from Kleinberg et al. [35] by allowing for heterogeneity in customer preferences. Despite its flexibility, the GSP choice model imposes some structure on the choice probabilities, and some examples are provided by the authors describing choice behaviors that do not belong to the GSP class. Ragain and Ugander [36] propose the Pairwise Choice Markov Chain model, where each alternative is represented as a node of a continuos time Markov Chain. Given an offer set, the choice probabilities are given by the stationary distribution of the sub-chain consisting of the nodes indexed by the available alternatives. Although the PCMC choice model is able to capture both substitution and halo effects, it obeys the axiom of uniform expansion introduced by Yellott [112]. The authors argue that such property may be desirable in the context of discrete choice modeling.

Universal discrete choice models. Some more general choice models have been proposed in the literature, which are able to represent *any* discrete choice function. In particular, Osogami and Otsuka [113] propose an extension of the MNL model aming at capturing high-order product interactions. They show that the resulting model can be represented as a Restricted Boltzman Machine (RBM), a probabilistic graphical model whose units are divided into two groups, visible and hidden. Visible units are used to encode a binary representation of the offer set and of a given choice, while hidden units learn a latent representation of the input. Given enough hidden units, these models can represent any sort of irrational behavior. An approach based on tree ensembles has recently been proposed by both Chen et al. [38] and Chen and Mišic [37], who show that any discrete choice model can be represented as a distribution over decision trees.

As previously mentioned, choice models with rather flexible structures pose some crucial challenges, whose solution greatly impacts the predictive accuracy of the trained choice models. In particular, one needs to balance between flexibility of the choice model, tractability of its estimation procedure, and risk of overfitting when limited amount of data is available. In this regard, it should be noted that our estimation procedure is non-parametric. Compared to the approach from Osogami and Otsuka [113], our model can adaptively increase the number of parameters in a data-driven way, as more data becomes available. This may help avoiding overfitting issues when only limited amounts of data, since a a model with a relatively small number of parameters will be learned in this case. The results from Berbeglia et al. [32] seems to confirm this claim, showing Rank-Based choice models are relatively data-efficient, offering good generalization even with limited amounts of transactions data. Also, our algorithm allows to include *both* rational and irrational behaviors in the estimation process, thus providing a general method, well suited to a wide range of real case studies. The same is not true for other irrational models such as Osogami and Otsuka [113], which should be used with care on datasets where it is reasonable to assume that customers do act rationally.

Chen and Mišic [37] and Chen et al. [38] tackle both the computational and generalization aspects by proposing regularization methods whose effect is to restrict the search space in a principled way. It may be argued, nevertheless, that less general choice models may be more effective in exploring search spaces that are smaller by definition, and that imposing some structure on the choice probabilities may provide important inductive bias to improve generalization over unseen offer sets when limited amount of data is available. This observation motivates the focus of this paper. In particular, we propose an estimation method for the Generalized Stochastic Preference choice model, which is flexible enough to subsume the RUM family of models and to capture halo effects, but still imposes some structure on the choice probabilities.

We conclude this section by mentioning an interesting line of work from the machine learning community, proposing general approaches based on neural networks to approximate the complex, high-order interactions among alternatives (see, e.g., Pfannschmidt et al. [114], Rosenfeld et al. [115], Mottini and Acuna-Agost [116]). Despite their flexibility, however, these models have only been applied to settings with product features and large number of training offer sets. Their adaptation to a setting close to ours, where no item featurization is given and the amount of offer sets seen at training time is relatively small, has not been explored yet and does not seem trivial.

6.3 The choice model

In this section, we review the Generalized Stochastic Preference model [2] and extend it by allowing for partial ordering and indifference sets. Consider a set of products $\mathcal{N} = \{0, ..., N-1\}$, with label 0 representing the *no-purchase* option. With a slight abuse of notation, let then σ denote both a subset of products in $\mathcal{N}$ and a linear order defined over such products, so that the rank (or position) of product j according to σ is given by $\sigma(j) \geq 1$.

Definition 4 *A Generalized Stochastic Preference [2] $C(\sigma, i)$, consists of a ranking of products $\sigma \subseteq \mathcal{N}$, and an index i, with $1 \leq i \leq |\sigma|$.*

Specifically, when faced with an offer set $S \subseteq \mathcal{N}$, a customer of type k, also referred to as $C_k(\sigma_k, i_k)$, picks the alternative ranked i^{th} in its subsequence σ_k that only contains items also available in S. Further, we define $\sigma_{k,S} \subseteq \sigma$ as the sequence of products obtained by removing from σ_k every product $j \notin S$. The customer will then choose product j^* so that $\sigma_{k,S}(j^*) = i$. If $|\sigma_{k,S}| < i$, the customer will leave without any purchase. The particular case of $i = 1$ corresponds to customers who always pick their favorite (i.e., highest ranked) product among the available ones. For this reason, we refer to customers $C(\sigma, 1)$ as *rational* behaviors, and to the index i of a generalized stochastic preference as its *irrationality level.* The GSP choice model is then defined by a probability distribution $\boldsymbol{\lambda} \in \mathbb{R}^K$ over K customer types $\{C_k(\sigma_k, i_k)\}_{k=1}^K$, so that the probability of choosing item j from assortment S is given by

$$P(j|S) = \sum_{k=1}^{K} \lambda_k \mathbb{1}\{\sigma_{k,S}(j) = i_k\}. \tag{6.1}$$

It should be noticed that, since every RUM choice model can be equivalently represented as a distribution over *rational* stochastic preferences (see, e.g., Block and Marschak [98]), the GSP choice model naturally subsumes the RUM family of models.

Kleinberg et al. [35] theoretically justify the use of "irrational", rank-based behaviors[2] in order

[2]In the work of Kleinberg et al. [35], the authors refer to *position-selecting choice functions*, whose defini-

to capture compromise effects. In particular, the authors assume alternatives can be mapped to a one-dimensional embedding (i.e., utility) representing the alternatives overall evaluation by the decision-maker. Such embeddings can be given by their price or by a possibly complex function of their features. Alternatives can then be ranked in this embedding space according to their evaluation. Then, choosing the best "compromise" corresponds to selecting the option with rank $2 \leq i \leq |S| - 1$, where S is the set of available alternatives. To better illustrate the practical implications of this argument, Table 6.2 exemplifies [2] how the GSP model can explain the results of the experiment from Simonson and Tversky [13] in Example 6.1.1.

Table 6.2 GSP model from Berbeglia [2] explaining the choice outcomes of Example 6.1.1. For each $\sigma_{k,S}$, we highlight in bold the chosen item $j : \sigma_{k,S}(j) = i_k$.

Customer Type			Probability	$S_1 = \{1,2\}$	$S_2 = \{1,2,3\}$
k	σ_k	i_k	λ_k	σ_{k,S_1}	σ_{k,S_2}
1	$(1,3,2)$	1	0.22	$(\mathbf{1},2)$	$(\mathbf{1},3,2)$
2	$(2,3,1)$	1	0.29	$(\mathbf{2},1)$	$(\mathbf{2},3,1)$
3	$(3,2,1)$	1	0.21	$(\mathbf{2},1)$	$(\mathbf{3},2,1)$
4	$(3,2,1)$	2	0.28	$(2,\mathbf{1})$	$(3,\mathbf{2},1)$

Table 6.3 Predicted shares of three camera models in choice scenarios S_1, where respondents must choose between alternatives $\{1,2\}$, and S_2, where option (3) is added to the offer set

		Predicted Share	
Model	Price ($)	S_1	S_2
(1) Minolta X-370	170	$\lambda_1 + \lambda_4 = .50$	$\lambda_1 = .22$
(2) Minolta 3000i	240	$\lambda_2 + \lambda_3 = .50$	$\lambda_2 + \lambda_4 = .57$
(3) Minolta 7000i	470	–	$\lambda_3 = .21$

When only products $\{1,2\}$ are offered, customer $k = 4$ chooses the cheapest one, i.e., option (1). According to the rational theory of choice, this would imply that option (1) has to be preferred to option (2) independently of the other products in the offer set. This, however, would be in contradiction with the data at hand. Irrational choice behaviors, on the other hand, can account for assortment-dependent effects. By introducing option (3) in the assortment, the same customer ends up choosing option (2), thus implying that (2) is preferred to (1) in this case. We refer to Appendix C.1 and Berbeglia [2] for more examples showing how the GSP choice model can reproduce several experiments from the literature on behavioral economics showing evidence of irrational choice behaviors.

tion is essentially equivalent to the one of (irrational) customer behaviors from Berbeglia [2].

From a modeling perspective, we note that including the 0 (i.e., no-purchase) option among the ranked alternatives has a useful implication in practice. In particular, contrary to the original formulation in Berbeglia [2], this allows us to capture violations of the regularity assumption also for the no-purchase option (we refer to Appendix C.2 for more details). Several studies, indeed, have shown that customers' willingness to purchase and overall satisfaction may decrease in the presence of too many alternatives among which a choice has to be made (see, e.g., Iyengar and Lepper [3], Schwartz [4]).

The estimation of the GSP choice model poses significant computational challenges, given that the space of rational customer types alone is factorially large. Estimation procedures developed for rational, rank-based models such as those from van Ryzin and Vulcano [30] and Bertsimas and Mišic [31] cannot be easily adapted to account for learning irrational preferences, nor does their scalability look promising to tackle the even bigger search space implied by the presence of irrational behaviors (see, e.g., [32, 73]). For these reasons, we decided to adopt the framework if partially-ranked preference sequences from Jena et al. [73] to represent generalized stochastic preferences. Besides providing a more intuitive, behavioral representation of an agent's decision process, partially-ranked preferences allow for fast estimation schemes and have been shown to generalize well on unseen offer sets. Starting from the observation that, for rational customer behaviors, low-ranked alternatives have a relatively low impact in explaining choice data, Jena et al. [73] propose to strictly rank only few, relevant alternatives for each preference list, while allowing for ties among the rest of them. Alternatives with the same rank may then be grouped into so-called *indifference sets.* We thus provide the following definition:

Definition 5 *A partially-ranked preference with irrationality $C(P(\sigma), I(\sigma), i)$, is defined by two sets of products $P(\sigma) \subseteq \mathcal{N}$ and $I(\sigma) \subseteq \mathcal{N} \setminus P(\sigma)$, respectively, and a linear ordering σ over the set of of alternatives $P(\sigma) \cup I(\sigma)$, so that $\sigma(j) = \sigma(j')$ for all $j, j' \in I(\sigma)$, and $\sigma(j) < \sigma(j')$ for all $j \in P(\sigma), j' \in I(\sigma)$.*

It follows from Definition 5 that a certain customer has no particular preference for products in $I(\sigma)$, which all have the same rank. Hence, we refer to $I(\sigma)$ as the *indifference set* of that customer type. Note, also, that the irrationality level of a partially-ranked preference is limited by $i \leq |P(\sigma)| + 1$, since alternatives in the indifference sets all have the same rank. For ease of notation, let $P_S(\sigma) = P(\sigma) \cap S$ and $I_S(\sigma) = I(\sigma) \cap S$ denote the strictly ranked preference list and the indifference set, respectively, obtained after removing from σ every product not available in a given offer set S. A customer $C(P(\sigma), I(\sigma), i)$ will then pick the alternative ranked i^{th} in $P_S(\sigma)$ if $i \leq |P_S(\sigma)|$, or an alternative chosen uniformly at random in $I_S(\sigma)$ when $|P_S(\sigma)| < i \leq |P_S(\sigma) \cup I_S(\sigma)|$. When $i > |P_S(\sigma) \cup I_S(\sigma)|$, the customer leaves without any purchase.

Table 6.4 Example: choice behavior of two customers $C_1\Big((2,3,5),\{1,4\},1\Big)$ and $C_2\Big((2,3,5),\{1,4\},2\Big)$ across different offer sets.

offer set		$P_S(\sigma)$	$I_S(\sigma)$	Choice$_{C_1}$	Choice$_{C_2}$
S_1	$\{2,5,1\}$	(2,5)	$\{1\}$	2	5
S_2	$\{2,1,4\}$	(2)	$\{1,4\}$	2	$\sim$ Unif$\{1,4\}$
S_3	$\{1\}$	()	$\{1\}$	1	0
S_4	$\{1,4\}$	()	$\{1,4\}$	$\sim$ Unif$\{1,4\}$	$\sim$ Unif$\{1,4\}$

While Section 6.4 elaborates on how to estimate such preference sequences from transaction data, Table 6.4 gives an example of the choice behavior of two hypothetical customers $C_1\Big((2,3,5),\{1,4\},1\Big)$ and $C_2\Big((2,3,5),\{1,4\},2\Big)$, who differ only by their irrationality level. As a consequence, for each offer set S, we have that $P_S(\sigma_1) = P_S(\sigma_2)$ and $I_S(\sigma_1) = I_S(\sigma_2)$. Specifically, the first row of Table 6.4 corresponds to the case where $i_1 = 1 \leq 2 = |P_S(\sigma_1)|$ and $i_2 = 2 \leq 2 = |P_S(\sigma_2)|$. The two customers will then select the items $j_1 = 2$ and $j_2 = 5$ with ranks 1 and 2, respectively, in $|P_S(\sigma)|$. For all the other assortments, however, we have that $i_2 = 2 > 1 \geq |P_S(\sigma)|$, hence Customer 2 will either pick an item uniformly at random from the indifference set (assortments S_2 and S_4) or leave without any purchase (assortment S_3). The same reasoning can be applied to obtain the choice of Customer 1 in the remaining assortments.

6.4 Estimation procedure

Jena et al. [73] have shown that *rational*, partially-ranked preferences can be efficiently learned from data, using a *Growing Preference Tree* (GPT) algorithm. In this section, we first review the GPT estimation framework, and then show how to extend the algorithm to additionally handle partially-ranked preferences with irrationality.

We assume that training data is available in the form of T observations $\mathcal{T} = \{(S_t, c_t)\}_{t=1}^{T}$ with S_t and c_t representing the offer set and the choice, respectively, that have been observed in period t. Let $\mathcal{S}_{train} = \{S_1, ..., S_M\}$ denote the collection of M offer sets over which choice data is available. We can further preprocess dataset $\mathcal{T}$ in order to obtain a vector of empirical probabilities $\boldsymbol{v} \in \mathbb{R}^{N \cdot M}$ so that, for each $j \in \mathcal{N}$ and $S \in \mathcal{S}_{train}$, the probability of item j

being chosen from offer set S is given by

$$v_{j,S} = \frac{\sum_{t=1}^{T} \mathbb{1}\{S_t = S, i_t = j\}}{\sum_{t=1}^{T} \mathbb{1}\{S_t = S\}}.$$

6.4.1 Non-parametric estimation framework

The GPT algorithm fits into the general column-generation framework proposed for the estimation of a general class of nonparametric choice models by van Ryzin and Vulcano [30]. In line with this framework, customer behaviors are represented as a choice matrix $\boldsymbol{A} \in \mathbb{R}^{(N\cdot M)\times K}$, encoding K behaviors for M offer sets, whose elements give the probability of customers choosing an item from a given offer set. In particular, based on the choice behaviors of a partially-ranked list with irrationality i defined in Section 6.3, the elements of the matrix $\boldsymbol{A}$ may be computed as follows:

$$A^k_{j,m} = \begin{cases} 1 & \text{if } j \in P_{S_m}(\sigma) \text{ and } j \text{ ranked } i^{th} \text{ in } P_{S_m}(\sigma), \\ \dfrac{1}{|I_{S_m}(\sigma)|} & \text{if } j \in I_{S_m}(\sigma) \text{ and } |P_{S_m}(\sigma)| < i \leq |P_{S_m}(\sigma) \cup I_{S_m}(\sigma)| \\ 0 & \text{otherwise.} \end{cases} \tag{6.2}$$

Given a distribution $\boldsymbol{\lambda} \in R^K$ over the customer types, the predicted probability $x_{j,m}$ of a random customer choosing alternative j from the offer set S_m is then given by $x_{j,m} = \sum_k A^k_{j,m}\lambda_k$. One can thus define the *best* distribution $\boldsymbol{\lambda}$, that is, the one for which the predicted probabilities are the closest to the observed ones, and obtain $\boldsymbol{\lambda}$ by solving the following optimization problem:

$$\min_{\boldsymbol{\lambda},\boldsymbol{x}} \quad \mathcal{L}(\boldsymbol{x},\boldsymbol{v}) \tag{6.3a}$$

$$\text{s.t.} \quad \boldsymbol{A}\boldsymbol{\lambda} = \boldsymbol{x} \tag{6.3b}$$

$$\mathbf{1}^T\boldsymbol{\lambda} = 1 \tag{6.3c}$$

$$\boldsymbol{\lambda} \geq 0. \tag{6.3d}$$

Here, $\mathcal{L}(\boldsymbol{x},\boldsymbol{v})$ can be any convex loss function measuring the distance between the predicted probabilities $\boldsymbol{x}$ and the observed ones $\boldsymbol{v}$. For example, one may minimize the L_1 error between the two probability distributions, in which case we have

$$\mathcal{L}(\boldsymbol{x},\boldsymbol{v}) = \sum_{S\in\mathcal{S}_{train}} \sum_{i\in S} |x_{i,S} - v_{i,S}|. \tag{6.4}$$

Minimizing the L_1 error generally leads to sparse models. Further, objective function (6.4) can be easily linearized (see, e.g., Bertsimas and Mišic [31]) and is therefore computationally amenable.

Another popular measure of the distance between two probability distributions is the Kullback-Leibler. It is strictly convex and leads to the same solution as Maximum Likelihood Estimation (see, e.g., Jagabathula and Rusmevichientong [33]). It is computed as

$$\mathcal{L}(\boldsymbol{x}, \boldsymbol{v}) = -\frac{1}{T} \sum_{S \in \mathcal{S}_{train}} T_S \sum_{i \in S} v_{i,s} \, log \frac{x_{i,S}}{v_{i,S}}, \tag{6.5}$$

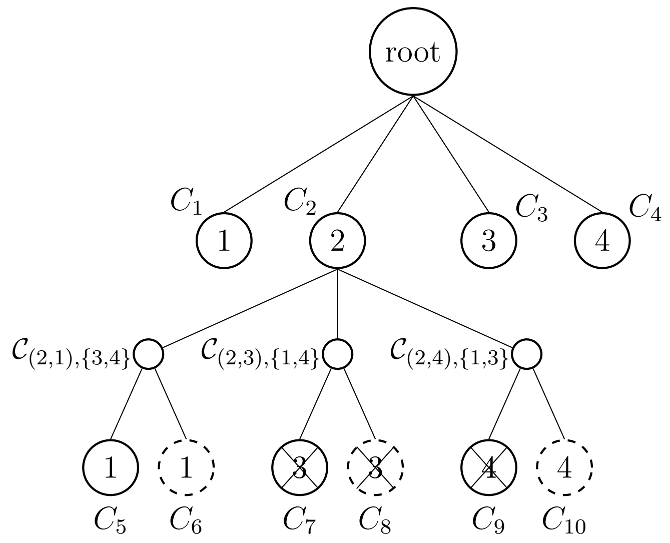
where T_S is the number of samples showing S as offer set.

6.4.2 Discovering new rational and irrational customer types

Solving problem (6.3) over the factorially large set of all possible customer types is not tractable. Hence, the master problem (6.3) is first initialized with a restricted set of customer behaviors and is solved to find the corresponding probability distribution $\boldsymbol{\lambda}$ that best fits the training data. In each further iteration, new relevant behaviors are discovered by solving a subproblem and added to the master problem, which then adjusts the probability distribution $\boldsymbol{\lambda}$ over the new set of behaviors. The algorithm terminates once a predefined stopping criteria is met.

In particular, let $\boldsymbol{\alpha} \in \mathbb{R}^{N \cdot M}$ and $\nu \in \mathbb{R}$ denote the dual variables associated with constraints (6.3b) and (6.3c), respectively. The customers (i.e., preference sequences) worth adding to the model in order to improve its fit of the data are those whose corresponding choice vector $\boldsymbol{a}$ (column of A), computed as in (6.2), has a negative (reduced) cost $c(\sigma) = -\boldsymbol{\alpha}^T \boldsymbol{a} - \nu$. While finding new preference sequences with negative costs generally tends to be computationally expensive, the GPT algorithm exploits the structure of partially-ranked preferences to efficiently identify such columns. Indeed, partially-ranked preferences allow for using an efficient tree-like data structure, where deeper levels correspond to behaviors with more refined ranked lists. Specifically, any sequence of products obtained from such a tree by traversing the path from the root to a given node, corresponds to the preference sequence $P(\sigma_k)$ of a certain customer C_k. It then follows that a behavior C_j is considered a sub-behavior of C_k, if $P(\sigma_j) = (P(\sigma_k), \ell)$ with $\ell \in I(\sigma_k)$. When searching for new customer behaviors, one may thus restrict the search for relevant customer types among the sub-behaviors of $\{C_1, ..., C_K\}$, at any given iteration.

To better illustrate how the search-tree is gradually explored during the GPT procedure, we report in Figure 6.1 the tree resulting from the initialization step, followed by one iteration of the GPT algorithm on a toy example, where the universe of products consists of four alternatives, i.e., $\mathcal{N} = \{1, 2, 3, 4\}$. The algorithm starts by initializing the tree with N *rational* customer types $C_1, \ldots, C_N$ such that $P(\sigma_k) = k$ and $I(\sigma_k) = \mathcal{N} \setminus \{k\}$. We then solve the restricted master problem (6.3) over this initial set of behaviors, in order to obtain a first distribution $\boldsymbol{\lambda}$ over the N customer types, and the values of the dual variables $\boldsymbol{\alpha}$ and ν. After the initialization, the generation of new candidate behaviors to include in the master problem at each future iteration requires three steps:



root
C_1
C_2
C_3
C_4
1
2
3
4
$\mathcal{C}_{(2,1),\{3,4\}}$
$\mathcal{C}_{(2,3),\{1,4\}}$
$\mathcal{C}_{(2,4),\{1,3\}}$
1
1
3
3
4
4
C_5
C_6
C_7
C_8
C_9
C_{10}

Customer	P(σ)	I(σ)	i	Init	Iter 1	
				$\boldsymbol{\lambda}$	$c(\sigma)$	$\boldsymbol{\lambda}$
C_1	(1)	$\{2,3,4\}$	1	0.1	-	0
C_2	(2)	$\{1,3,4\}$	1	0.7	-	0.2
C_3	(3)	$\{1,2,4\}$	1	0.2	-	0.3
C_4	(4)	$\{1,2,3\}$	1	0	-	0
C_5	$(2,1)$	$\{3,4\}$	1	-	**-1**	0.1
C_6	$(2,1)$	$\{3,4\}$	2	-	**-4**	0.3
C_7	$(2,3)$	$\{1,4\}$	1	-	0.2	-
C_8	$(2,3)$	$\{1,4\}$	2	-	-0.1	-
C_9	$(2,4)$	$\{1,3\}$	1	-	0.1	-
C_{10}	$(2,4)$	$\{1,3\}$	2	-	**-3**	0.1

Figure 6.1 (Left) Search-Tree of GPT for finding new behavior, on a toy example with four products, after two iterations. A path in the tree corresponds to a sequence of strictly ranked products. Dashed nodes correspond to irrational behaviors. (Right) The explicit behaviors description, with corresponding probabilities and costs at a given iteration.

1. **Sampling:** We select γ behaviors from the existing nodes via random sampling according to probability distribution $\boldsymbol{\lambda}$. In the example in Figure 6.1, we have sampled $\gamma = 1$ behavior in the first phase: namely, C_2 (i.e., $k = 2$ with probability $\lambda_2 = 0.7$).

2. **Sub-behavior generation:** For every sampled behavior, we first generate $|I(\sigma_k)|$ *rational* sub-behaviors (C_5, C_7 and C_9 in the example). These new behaviors $(P(\sigma), I(\sigma), 1)$ are obtained by removing one item from the indifference set of the parent node (C_2 in the example) and adding it to the end of its strictly ranked preference sequence ($P(\sigma_2)$). Such rational sub-behaviors (with irrationality level $i = 1$) correspond to those also added by Jena et al. [73]. We here extend this approach by additionally generating the *irrational* counterparts with irrationality levels i with $1 < i \leq |P(\sigma)| + 1$ (C_6, C_8 and C_{10} in the example).

3. **Sub-behavior selection:** We then compute the reduced costs $c(\sigma)$ for each candidate sub-behavior and select the δ behaviors with the best (i.e., smallest) reduced costs, where δ is a pre-specified hyper-parameter (in the above example, $\delta = 3$, selecting customers types C_5, C_6 and C_{10}). The nodes corresponding to the remaining customer types are pruned from the search-tree (nodes C_7, C_8 and C_9 in the example).

Identifying relevant customer behaviors based solely on their reduced costs may not be robust in general when dealing with irrational customer behaviors. We elaborate more on the topic in Section 6.4.3 where, based on behavioral considerations, we propose a superior selection criteria for the identification of relevant customer types.

Observe that customers that differ only in their irrationality level (such as C_5 and C_6 in our example) generate the same sets of sub-behaviors. Hence, only one of these nodes (either C_5 or C_6 in our example) has to be considered to evaluate and generate further sub-behaviours. We therefore consider only one node among the set of behaviors $\mathcal{C}_{P(\sigma),I(\sigma)} = \{C_k(P(\sigma_k), I(\sigma_k), i_k) : P(\sigma_k) = P(\sigma) \text{ and } I(\sigma_k) = I(\sigma), k = 1, ..., K\}$, with probability $\tilde{\lambda}_{P(\sigma),I(\sigma)} = \sum_{k:C_k \in \mathcal{C}_{P(\sigma),I(\sigma)}} \lambda_k$. Considering the example in Figure 6.1, at the second iteration the preference list $((2,3), \{1,4\})$ would then be sampled with probability $\tilde{\lambda}_{((2,3)\{1,4\})} = \lambda_5 + \lambda_6 = 0.4$.

We highlight two major advantages of the exploration strategy employed by the GPT algorithm:

1. The number of strictly ranked objects increases in an adaptive, data-driven way, with more refined preference lists added only when needed. Besides allowing to avoid the computational burden of strictly ranking all the products in a preference list, Jena et al. [73] show that the explanatory power of indifference sets tends to improve generalization on new offer sets.

2. Since the irrationality level i of a partially-ranked behavior is bounded by the number of strictly ranked products, i.e., $1 \leq i \leq |P(\sigma)|$, the GPT search procedure prioritizes customers with low irrationality levels. This seems to be a behaviorally-plausible inductive bias, which may reduce the risk of overfitting, especially when only a limited amount of data is available.

Computational complexity. As described above, each iteration of the GPT procedure involves sampling γ behaviors $C_k(P(\sigma_k), I(\sigma_k), i_k)$ and, for each them, generating $|I(\sigma_k)| \cdot (|P(\sigma_k)| + 1)$ sub-behaviors, i.e., one for each item $j \in I(\sigma_k)$ and irrationality level

i, with $1 \leq i \leq |P(\sigma_k)| + 1$. The reduced cost $c(\sigma)$ of a sub-behaviour σ can be obtained in $O(M)$ [73] from the reduced cost of its parent node. Therefore, computing the reduced costs of all rational and irrational candidate sub-behaviors at a given iteration has a complexity of $O(N^2M)$ (compared to $O(NM)$ for the rational approach). While this is the theoretical worst-case complexity, our experiments in Appendix C.4.1 show that the GPT tends to produce relatively short preference lists $P(\sigma)$, ranking as few as 3 products, on average. Further, Jena et al. [73] showed that even on large (rational) instances with up to 1,000 products, the length of the produced strict preference lists tends to be similarly small, and rather independent from the number of products. The computational burden of each GPT iteration is therefore significantly mitigated in practice, since is tends to generate few irrational behaviors.

6.4.3 A new dominance rule to select relevant customer types

In this section, we elaborate on how to identify new customer behaviors that improve the fit to the training data and are more likely to generalize well on test data. The *de facto* scoring method in column generation, and therefore for evaluating the quality of candidate behaviors in the GPT estimation procedure, is exclusively based on their reduced costs $c(\sigma)$. This is based on the hypothesis that a parsimonious model tends to generalize well on unseen offer sets, which is widely adopted in the literature (see, e.g., van Ryzin and Vulcano [30], Bertsimas and Mišic [31], Jena et al. [73]). However, when dealing with irrational behaviors, such criterion may lead to capturing an excessive number of spurious interactions among products, especially in the case of scarce data availability.

In the following, we argue that the use of customer behaviors with small numbers of strictly ranked products, and, as a consequence, larger indifference sets, may drastically reduce the risk of overfitting. Using rational behaviors only, this is naturally achieved by the design of the GPT algorithm, which starts by generating behaviors with small numbers of strictly ranked products. While, in the rational case, strictly ranking only a few products with respect to the total number of existing ones (e.g., 5 out of 100 products) may be sufficient to reduce the risk of overfitting, this may not be the case when considering irrational behaviors. In fact, here, the added risk of overfitting increases quickly with the number of possible interactions within the sequence of strictly ranked products.

As an example, consider the case in which we aim to capture the positive relation of item 1 on item 2 in a total universe of $N = 5$ products (for simplicity, we here do not use the no-purchase option in the preference sequences or the assortments), based on the observed choices of product 3 from assortment $S_1 = \{2, 3\}$ and product 2 from assortment $S_2 = \{1, 2, 3\}$. When estimating the choice model, several irrational preference sequences may fit these transactions, for example, either $C\Big((1,2),\{3,4,5\}),2\Big)$ or $C\Big((1,2,3,4,5),\{\},2\Big)$. The former (with few strictly ranked products) is clearly the more precise one, and is less prone to overfitting, while the latter is an example of a more general one, prone to a higher risk of overfitting. While the latter sequence seems to have unnecessarily many strictly ranked products, in practice, when dealing with limited transaction data, such sequences may be selected due to their lower reduced costs. A choice model using such a sequence may, depending on the unseen offer sets, extrapolate up to $|P| \cdot (|P|-1)/2$ possible pairwise interactions, which are illustrated in Table 6.5. Note that all except for one of these interactions are spurious and therefore overfit the training data. In contrast, the former sequence with only two strictly ranked products "deactivates" the influence of product 1 on any product other than 2. While

Table 6.5 Some of the positive interactions $j \to j'$ implied by a single customer behavior $C\Big((1,2,3,4,5),2\Big)$. In particular, by considering the offer set $S_2 = S_1 \cup \{j\}$, customer's choice changes in favour of product j'.

Positive interaction	S_1	Choice$_{S_1}$	S_2	Choice$_{S_2}$
$1 \to 2$	$\{2,3\}$	3	$\{1,2,3\}$	2
$1 \to 3$	$\{3,4\}$	4	$\{1,3,4\}$	3
$1 \to 4$	$\{4,5\}$	5	$\{1,4,5\}$	4
$1 \to 5$	$\{5\}$	0	$\{1,5\}$	5
$2 \to 3$	$\{3,4\}$	4	$\{2,3,4\}$	3
$2 \to 4$	$\{4,5\}$	5	$\{2,4,5\}$	4
$2 \to 5$	$\{5\}$	0	$\{2,5\}$	5
$3 \to 4$	$\{4,5\}$	5	$\{3,4,5\}$	4
$3 \to 5$	$\{5\}$	0	$\{3,5\}$	5
$4 \to 5$	$\{5\}$	0	$\{4,5\}$	5

it is, of course, not possible to know beforehand which products really do interact which each other, it becomes immediate that, in an effort of reducing the risk of overfitting, it is desirable to prioritize behaviors with a small number of strictly ranked products.

The number of spurious interactions to which an irrational customer behavior may lead to can be quantified exactly, as demonstrated in the following observation.

Observation 1 ***(Number of spurious positive interactions)****: Let $C(P(\sigma), I(\sigma), i)$ be a partially-ranked preference sequence with strictly ranked products $P(\sigma)$, indifferent set $I(\sigma)$ and level of irrationality i. Depending on the unseen offer set, a choice-model based on C may extrapolate up to $\sum_{j=i-1}^{|P(\sigma)|-1} \binom{j}{i-1}$ positive interactions of degree i.*

Proof: see Appendix C.5.

Observation 1 implies that the number of spurious interactions, leading to potential overfitting, does not necessarily grow with a higher level of irrationality, but rather with the number of strictly ranked products. We note that this observation is not limited to our definition of partially-ranked preference sequences, but generally applies to the case of behaviors defined by the Generalized Stochastic Preference model from Berbeglia [2].

We now propose a new dominance rule to select the candidate behaviors to be included in the master problem (6.3). The rule aims at striking a delicate balance between the number of strictly ranked products and the reduced costs of a candidate behavior:

1. For every behavior σ belonging to the the set of candidate behaviors at a given iteration, we use equation (6.2) to compute its choice vector representation $\boldsymbol{a}$, and its cost $c(\sigma) = -\boldsymbol{\alpha}^T\boldsymbol{a} - \nu$;

2. We then lexicographically sort the candidate behaviors: first in increasing order of their number of strictly ranked products, i.e., $|P(\sigma)|$, and second in non-increasing order of their reduced costs $c(\sigma)$. This sorting operation induces a ranking π among the candidate behaviors;

3. Ranking π itself does not guarantee that its highest-ranked candidates have negative reduced costs. While it is reasonable to add some candidates with non-negative costs (which have shown in experiments to be beneficial for future iterations), we have to ensure also adding candidates with negative costs in order to improve the model fit. We therefore start considering candidates in ranking π at the highest rank corresponding to a behavior with negative cost: $\underline{\pi}= \arg\min_k\{\pi(\sigma_k)|c(\sigma_k) < 0\}$. We then select all candidate behaviors $\{k|\underline{\pi} \leq \pi(\sigma_k) \leq \underline{\pi} + \delta - 1\}$, for a pre-defined δ.

As will be shown in Section 6.5, the use of this selection rule improves the predictive performance of the irrational GPT algorithm, particularly on the real-world data set.

Table 6.6 exemplifies the selection process according to each of the two possible selection criteria for seven candidate behaviors. The first four lines show, respectively, the behavior id, the number of strictly ranked products, the cost of each behavior and the resulting ranking π. Using cost $c(\sigma)$ as selection critera would lead to selecting customers with smallest cost $c(\sigma)$, namely C_1, C_4 and C_7. However, notice that customer C_7 strictly ranks a relatively high number of products and, based on Observation 1, may imply a high number of spurious interactions among products. By using our proposed selection criterion, instead, we select

Table 6.6 Difference in selected columns using a criterion based solely on the cost $c(\sigma_k)$, and one where columns are first ordered based on the number of strictly ranked products $|P(\sigma_k)|$

k	3	1	4	5	2	6	7
$\|P(\sigma_k)\|$	1	2	2	2	3	3	4
$c(\sigma_k)$	0.2	-2	-1	0.01	-0.1	3	-3
$\pi(\sigma_k)$	1	2	3	4	5	6	7
Criterion	Entering Columns						
$c(\sigma_k)$		✓	✓				✓
$(\|P(\sigma_k)\|, c(\sigma_k))$		✓	✓	✓			

behaviors with only two strictly ranked products. In particular, assuming $\delta = 3$, we have $\underline{\pi} = 2$ and will select behaviors C_1, C_4 and C_5.

We conclude this section highlighting connections between our proposed dominance rule and *consideration sets*, a well known concept in the Marketing literature (see, e.g., Hauser et al. [117]). This concept refers to the fact that consumers usually pay attention to only a small subset of all existing items (either due to a limited attention budget or to avoid the cognitive burden of searching through a possibly very large set of products), and will select an item from this subset.

While the estimation of consideration sets is object of a separate branch of literature (see, e.g., Aouad et al. [87], Jagabathula et al. [118]), our notion of strictly ranked sets approximate consideration sets, while still attributing a non-zero probability to products in the indifference set. It has also been shown in the literature on brand choice that the size of consideration sets tends to be relatively small, usually consisting of 2 to 5 brands (see, e.g., Hauser and Wernerfelt [119]). This suggests, from a behavioral point of view, that prioritizing behaviors with a small number of strictly ranked products (and therefore lower orders of interactions among products) may provide good inductive bias for generalizing well on unseen offer sets.

6.5 Computational results

In this section, we report the results of our experiments on both synthetic and real datasets. The goal is to understand whether irrational, partially-ranked behaviors can improve predictive accuracy on new offer sets. In all our experiments, we compare three variants of the partially-ranked choice model estimated using GPT, namely GPT-R, which corresponds to the approach of Jena et al. [73], thus restricting the space of customer behaviors to the rational ones only, and its extensions GPT-I and GPT-IC, which include irrational behaviors in the estimation process. Of these, the former selects customer behaviors based solely on their costs, while the latter use the dominance rule proposed in this work, aiming at prioritizing customer behaviors with small *consideration sets*, i.e., with a small number of strictly ranked products, and thus focusing on sparse, low-order interactions among products. We further compare the GPT-based approaches with three benchmarks: the enumerative rank-based choice model (RB-R) with fully-ranked lists, obtained by enumerating all possible preferences (i.e., permutations over products)[3] , the pairwise choice markov chain (PCMC) proposed by Ragain and Ugander [36], and the Halo-MNL choice model from Maragheh et al. [15]. Section 6.5.1 focuses on the generalization performances of the various approaches on a set of synthetic instances. In Section 6.5.2, we test the models on the IRI Academic dataset [120], consisting of transaction data from a large grocery store chain.

Estimation of the choice models. Following Jagabathula and Rusmevichientong [33], we train RB-R by minimizing the average Kullback-Leibler (KL) divergence (6.5) between predicted probability distributions and the empirical ones over training offer sets. All the other approaches are trained by Maximum Likelihood Estimation. As already observed, it is well known that minimizing the KL divergence is equivalent to maximum likelihood estimation in terms of optimal solution retrieved (see, e.g., Jagabathula and Rusmevichientong [33]). For the GPT-based approaches, we stop the training procedure when the difference in the log-likelihood of the data at two consecutive iterations is statistically insignificant, as proposed by van Ryzin and Vulcano [30][4] , or when no negative cost column has been found at a given iteration. Further, we set the hyperparameters $\gamma = 10$, and $\delta = 20$ (as in Jena et al. [73]), which denote the number of behaviors to sample and the number of best ones to add to the master problem at each iteration, respectively. For the estimaton of PCMC, we used the code provided by the authors[5]. We refer the reader to Appendix C.3 for more details on the implementation of the PCMC choice model.

6.5.1 Numerical results on synthetic instances

[3]Let m denote the maximum number of products missing from any assortment. As noted in Jagabathula and Rusmevichientong [33] and Honhon et al. [67], only $O(N^m)$ permutations of $m+1$ products need to be generated, since products with rank greater than $m+1$ will never be chosen.

[4]Let $\mathcal{L}^k$ and $\chi^2(\beta)$ denote the log-likelihood at iteration k and the critical value of the chi-squared distribution with β degrees of freedom, respectively. We stop the GPT-procedure when $-2(\mathcal{L}^k - \mathcal{L}^{k+1}) < \chi^2(\beta)$, where β is replaced by the difference in the number of parameters (i.e., behaviors) between the two iterations.

[5]Code available at https://github.com/sragain/pcmc-nips.

We start this section by describing our data generation procedure. We then proceed by characterizing the irrationality level of the generated instances, and by reporting the generalization performance of the various approaches on such instances.

6.5.1.1 Data Generation.

We generate choice data samples according to two ground-truth models, specifically the Halo-MNL model proposed by Maragheh et al. [15] and the GSP model. Both of them allow us to control the amount of irrationality resulting in the generated instances and to investigate its impacts on the performance of the various approaches. For each ground truth model, instances were generated as follows:

- **Halo-MNL**: this choice model is parametrized by a pairwise interaction matrix U, whose diagonal terms u_{ii} represent the item-specific utilities. Given the offer set S, the overall probability of choosing product i is given by

$$P(i|S) = \frac{\exp(u_{ii} + \sum_{k \notin S} u_{ki})}{\sum_{j \in S} \exp(u_{jj} + \sum_{k \notin S} u_{kj})} \ .$$

  It is easy to see that by setting to zero the off-diagonal terms of matrix U, we obtain an MNL model. Following Chen and Mišic [37], we draw the elements $u_{ii} \sim \text{Unif}[-1, 1]$. We vary the irrationality of the instances by varying the number of pairwise interactions. Specifically, we generate instances where 0%, 10% and 25% of the couples present a positive interaction, obtained by setting the corresponding off-diagonal terms to -1. We simulate both symmetric halo effects, where two products increase each other's attractiveness, and asymmetric halo effects, also known as *decoy* effects, where only one of two products benefits from the presence of the other (the decoy) in the offer set. In order to investigate more complex interaction scenarios, we generalize the Halo-MNL model so as to include multiple customer segments, whose probability is drawn uniformly from the unit simplex. In our experiments, we have used either one or ten customer segments. We note again that when setting the number of pairwise interactions to zero, we end up obtaining rational instances generated under MNL and MMNL ground-truth models, depending on the number of customer segments, 1 and 10, respectively.

- **GSP**: We remind from Section 6.3 that a generalized stochastic preference is defined as $C(\sigma, i)$, where σ is a ranking over the N alternatives, and i is the irrationality level of the customer type. Instances generated under this ground-truth model contain either 10 or 100 customer types, whose probabilities are randomly drawn from the unit simplex. For each instance, we consider 10%, 20% or 50% of the customer types as irrational, meaning their index i is greater than one. Specifically, the irrationality level i of each of these customer types was randomly chosen in $\{1, 2, ..., i_{max}\}$. We used $i_{max} = 1, 5,$ and 9 to simulate various levels of irrationality. Rational instances have been obtained by setting the percentage of irrational behaviors to zero.

In all the experiments reported in this section, we used a number of products $N = 10$, one of which represents the no-purchase option. For each ground-truth model, we generate either $3,000$ or $50,000$ transactions, for a total of 10, 20 or 50 training offer sets. This simulates different amounts and diversity of training data. When using 50,000 transactions, in particular, the goal is to simulate the scenario in which we train the models based on empirical probabilities that are close to the true ones (i.e., those from the ground-truth model), and the effect of any sampling noise becomes negligible. This corresponds to the setting already used, for example, in Bertsimas and Mišic [31] and Chen and Mišic [37], where choice models are trained on ground truth probabilities. We further assume that the number of transactions is equally distributed among the training offer sets, which all have dimension $|S| \geq 3$ and contain the no-purchase option.

6.5.1.2 Loss of Rationality.

We first investigate the level of *irrationality* present in the generated instances. In line with the methodology proposed by Jagabathula and Rusmevichientong [33], we fit the enumerative rank-based choice model, RB, to all our training instances. It is well known that any RUM choice model can be equivalently represented as a probability distribution over rankings of alternatives [98]. Thus, by fitting such model to a given instance, the resulting objective function indicates what the authors define as the *Loss of rationality* (LoR) of that instance, which can be interpreted as a measure of the minimum amount of choice data that cannot be explained by using any choice model belonging to the RUM family. Figure 6.2 reports the LoR value distributions over instances grouped by category (Rational and Irrational), ground-truth models (Halo-MNL and GSP) and number of customer types (in parenthesis). As already mentioned, rational instances for Halo-MNL(1) and Halo-MNL(10) correspond to instances generated under MNL and MMNL ground-truth models, respectively. Also, rational GSP ground-truth models are equivalent to Rank-Based models with the same number of preference lists.

It is interesting to note that the aggregation of a high number of irrational customer types seems to result into a rational choice behavior at the population level (see Halo-MNL(10)

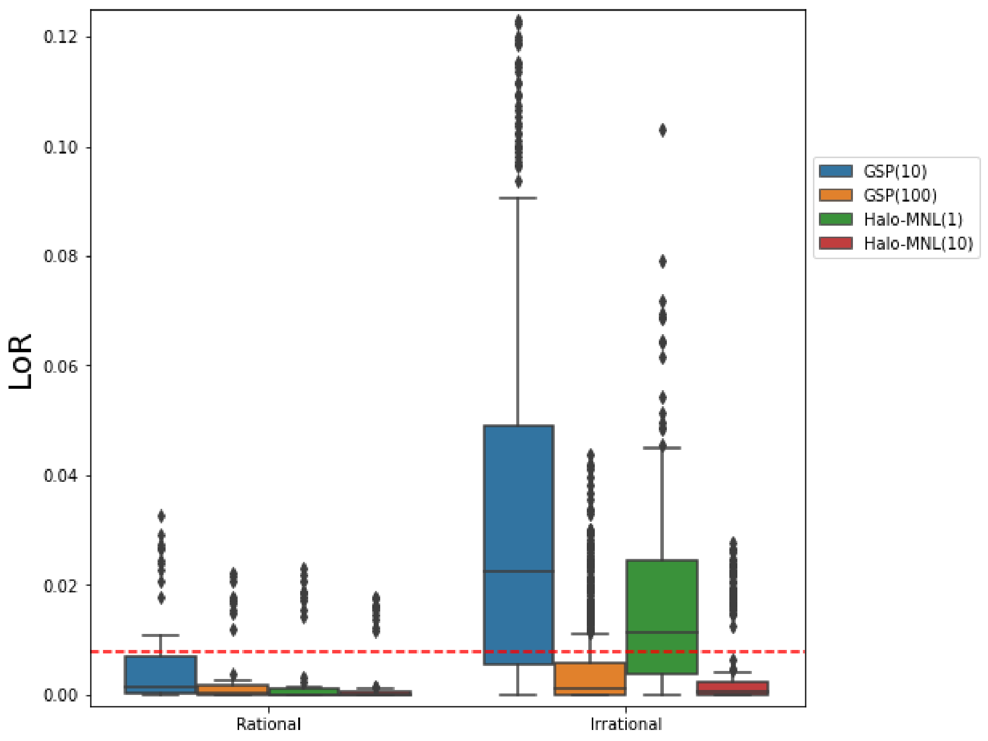


Figure 6.2 Distributions of *Loss of Rationality* for generated instances, grouped by ground-truth models and number of customer behaviors.

and GSP(100) in Figure 6.2), given that the individual, complex buying behavior becomes less apparent, and the collective (aggregated) transactions are easier approximated by simple choice rules. This is also connected to what Berbeglia et al. [32] refers to as degree of *consistency*. Instances with many customer types are said to be "less consistent", since the probability of being of a certain customer type is relatively low, and have been found to generalize better.

Figure 6.2 also reports a red dashed line, corresponding to a Loss of rationality of 0.008, which visually separates the generated instances based on their irrationality level. Essentially, rationally-generated instances tend to fall below this threshold, but also the irrationally-generated ones in which many customer types are aggregated. In the following, we interpret this value as a threshold to understand whether an instance contains significant amount of irrational choice behaviors. In Appendix C.4.2, we further show that the LoR of a given instance can be impacted by other factors as well, such as the number of choice samples and offer sets available for training.

6.5.1.3 Generalization performances.

We now focus on the generalization performance of the various approaches when tested on new offer sets. This has been measured in terms of average L_1 error between the predicted probability distribution $\boldsymbol{x}$ and the ground-truth probability distribution $\boldsymbol{v}$ on new offer sets, and has been computed as

$$L_1(\boldsymbol{x}, \boldsymbol{v}) = \frac{1}{|\mathcal{S}_{test}|} \sum_{S \in \mathcal{S}_{test}} \sum_{i \in S} \Big| x_{i,S} - v_{i,S} \Big|, \tag{6.6}$$

where $\mathcal{S}_{test}$ is the collection of *all* possible offer sets that have not been used for training[6]. As noted in Ragain and Ugander [36], equation (6.7) can be interpreted as the expected L_1 prediction error given a randomly drawn offer set.

Table 6.7 reports the L_1 test errors of each approach on sets of instances grouped by ground-truth models and number of customers types, indicated in parenthesis.

We first focus on the set of irrational instances. It is possible to observe how the performance of the rational choice models, RB-R and GPT-R, deteriorates as the Loss of Rationality increases. Specifically, this is the case for Halo-MNL(1) and GSP(10) instances where, among the rank-based approaches, GPT-IC and GPT-I offer the best predictive accuracy, respectively. GPT-IC favourably compares also to the Halo-MNL for small enough percentage of positive, pairwise interactions among products. On average, however, the Halo-MNL choice model manages to well approximate the data-generating process on Halo-MNL(1) instances, therefore outperforming all other approaches. We notice that GPT-IC tends to outperform GPT-I on all classes of instances, with the exception of those in GSP(10). On these instances, GPT-IC does actually not improve over GPT-R, unless there is a high percentage of irrational behaviors. Such results are in line with the behavioral assumption intrinsic to GPT-IC, which we proposed to explicitly prioritize sparse, low-order interactions among products, and may thus fail to generalize well on those scenarios where customers consistently (i.e., with high probability) show high orders of substitution and halo effects among products. However, we deem such interactions less likely to happen in practice. Our results on real-world data (see Section 6.5.2) seem to confirm this claim. The good median and maximum test errors of GPT-IC further confirm the robustness of the underlying selection criterion to identify customer behaviors that tend to generalize well.

We now discuss the results for rational instances, where the flexibility of irrational approaches may increase the risk of overfitting (see also Appendix C.4.1) when compared to GPT-R. All

[6]The total number of offer sets with dimension $3 \leq |S| \leq 10$ and containing the no-purchase option is equal to 502. Hence, given $M = 10, 20$ or 50, $|\mathcal{S}_{test}| = 502 - M$.

Table 6.7 L_1 test errors for each approach on Irrational instances grouped by ground-truth model, number of customer types(in parenthesis) and percentage of irrational behaviors/interactions used to generate the data.

	% irrat	LoR	RB-R	GPT-R	GPT-I	GPT-IC	PCMC	Halo-MNL
Irrational instances								
Halo-MNL(1)	10	0.0103	0.2176	0.2305	0.1693	**0.1648**	0.2606	0.2028
Halo-MNL(1)	25	0.0244	0.3229	0.3280	0.2843	0.2801	0.3804	**0.2370**
	(all) Mean	0.0173	0.2702	0.2792	0.2268	0.2224	0.3205	**0.2199**
Halo-MNL(10)	10	0.0035	0.1462	0.1128	0.0995	**0.0979**	0.2196	0.1917
Halo-MNL(10)	25	0.0044	0.1689	0.1574	0.1421	**0.1378**	0.2518	0.2038
	(all) Mean	0.0040	0.1575	0.1351	0.1208	**0.1179**	0.2357	0.1977
GSP(10)	10	0.0171	0.2441	0.2208	**0.2171**	0.2337	0.4271	0.6258
GSP(10)	20	0.0286	0.2805	0.2625	**0.2538**	0.2675	0.4624	0.6868
GSP(10)	50	0.0573	0.3735	0.3651	**0.3388**	0.3563	0.5314	0.8100
	(all) Mean	0.0343	0.2994	0.2828	**0.2699**	0.2858	0.4736	0.7075
GSP(100)	10	0.0034	0.1679	**0.1409**	0.1440	0.1438	0.2890	0.2702
GSP(100)	20	0.0043	0.1801	**0.1573**	0.1617	0.1587	0.3009	0.2737
GSP(100)	50	0.0082	0.2160	0.2044	0.2046	**0.2030**	0.3394	0.3154
	(all) Mean	0.0053	0.1880	**0.1675**	0.1701	0.1685	0.3098	0.2865
(all) Mean		0.0170	0.2345	0.2196	**0.2058**	0.2096	0.3567	0.4083
(all) Median		0.0046	0.2083	0.1882	0.1772	**0.1734**	0.3279	0.3078
(all) Max		0.2343	0.9454	0.7877	0.8087	**0.7835**	1.0035	1.5872
Rational instances								
MNL	-	0.0034	0.1227	**0.0916**	0.0933	0.0918	0.1404	0.1772
MMNL	-	0.0027	0.1294	**0.0724**	0.0762	0.0760	0.1529	0.1823
RB(10)	-	0.0060	0.1622	**0.1468**	0.1636	0.1812	0.3841	0.5539
RB(100)	-	0.0033	0.1559	**0.1251**	0.1317	0.1282	0.2793	0.2498
(all) Mean		0.0039	0.1425	**0.1090**	0.1162	0.1193	0.2392	0.2908
(all) Median		0.0003	0.1361	**0.1012**	0.1097	0.1088	0.2310	0.2291
(all) Max		0.0327	0.4697	0.4117	0.4211	**0.4075**	0.6795	1.2308

GPT-based algorithms favorably compare with RB-R, confirming the results of Jena et al. [73] on the generalization power of partially-ranked lists and that enumerating all possible fully-ranked preferences for training the RB model increases the risk of overfitting the training set [30]. This supports the hypothesis that adding only relevant types to the estimated choice model is crucial for its generalization performance. Also note that the same relative performance between GPT-I and GPT-IC observed on irrational instances can be noticed in the rational case as well. Specifically, GPT-IC outperforms GPT-I, on average, on all instances with the exception of RB(10) ones, i.e., those where customers consistently exhibit high order of substitutions among products. Finally, we emphasize that both irrational GPT variants significantly outperform the PCMC and Halo-MNL models on such rational instances.

Impact of the amount of available data We now test the robustness of the various approaches under different amounts of training data. In particular, Figure 6.3 and Figure 6.4 report, for each approach, the corresponding average test error over irrational and rational instances, respectively, under different data-regime settings "T-M", where T represents the number of transactions and M the number of assortments available for training. Each plot also reports, on the top axis, the best performing model under each data-regime scenario.

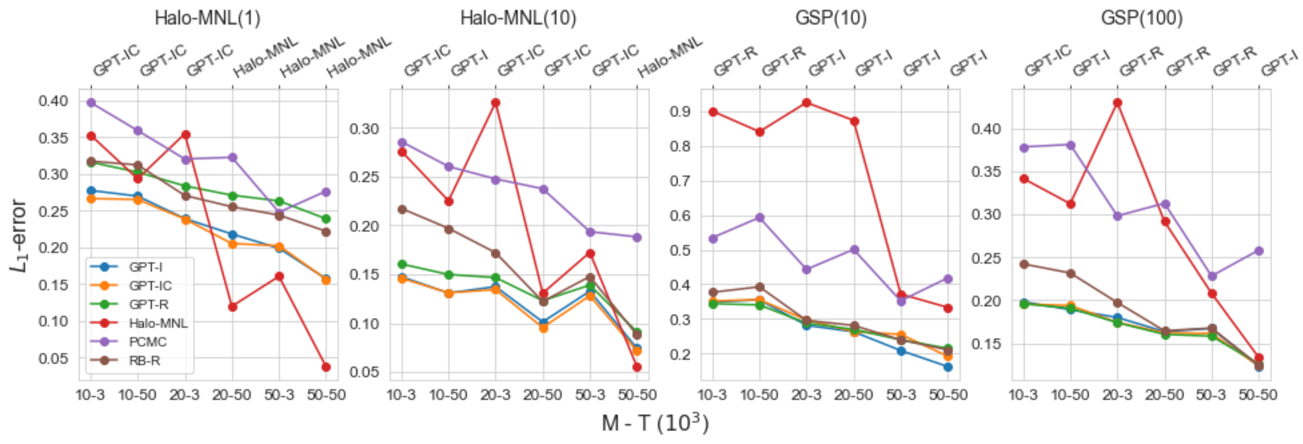


Figure 6.3 Average L1 test error of the various approaches on Irrational instances. The bottom axis reports the number of unique assortments M and transactions T (in 10^3 units) available for training. The top axis shows, for each data-regime setting, the best performing approach.

Both figures exhibit common trends in the the relative performance of the approaches. In particular, the Halo-MNL model tends to require significant amounts of data in order to become competitive with (or outperform) the rank-based approaches. These, in contrast, show relative stable performance and steadily improve with more available training data. They also consistently outperform the PCMC model.

Among the rank-based approaches, in line with our discussion of Table 6.7, we observe that GPT-IC tends to perform the best on Halo-MNL instances. Also, while being competitive on GSP(100) and RB(100) instances, it is generally outperformed by other rank-based approaches on GSP(10) and RB(10) instances, which are characterized by higher orders of interactions among products.

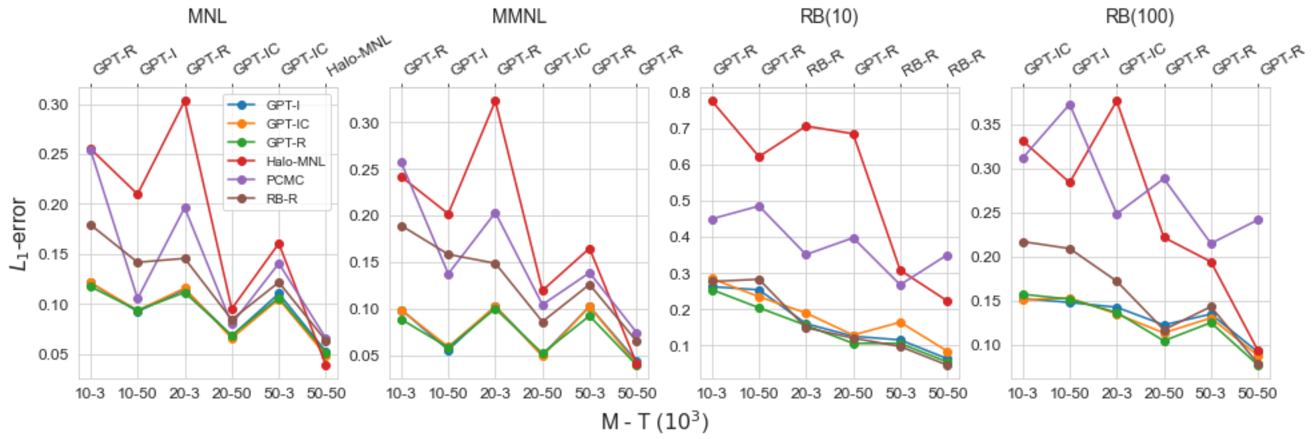


Figure 6.4 Average L1 test error of the various approaches on rational instances. The bottom axis reports the number of unique assortments M and transactions T (in 10^3 units) available for training. The top axis shows, for each data-regime setting, the best performing approach.

We conclude this section by referring the interested reader to Appendix C.4 for additional numerical results. In particular, Appendix C.4.1 provides statistics describing the choice models learned by GPT-based approaches on classes of instances characterized by different levels of irrationality. The results therein also highlight the computational effectiveness of GPT-based approaches, which can be estimated in less than two seconds, on average, on our synthetic instances. In Appendix C.4.3, we provide a more refined view of the impact of the irrationality level of GSP customer types on the predictive accuracy of the various approaches. Finally, Appendix C.4.4 investigates the generalization performance of the implemented algorithms on Halo-MNL instances separated by type of positive interaction, i.e., symmetric or asymmetric, in the ground-truth model.

6.5.2 Numerical results on the IRI Academic dataset

In this section, we test all approaches on the IRI Academic data-set [120], a real-world data-set from the retail sector. The data consists of transactions from grocery and drug store chains located in 47 markets in the USA. We consider transactions corresponding to 29 product categories in total, which we report in Table 6.8.

6.5.2.1 Dataset pre-processing.

Each transaction record in the dataset contains informations regarding the week and store in which the transaction occured, and the purchased item, uniquely identified by its Universal Product Code (UPC). In order to tackle both the sparsity and large volume of the data, we follow the same pre-processing steps outlined in Jagabathula and Rusmevichientong [33]. Specifically, we start by considering the subset of transactions corresponding to the first two weeks of year 2007. We then aggregate products by their UPC-vendor code. In other words, transactions where items with the same UPC-vendor code were bought, are attributed to the same *vendor-item*. This kind of aggregation is common in the marketing literature in order to reduce the sparsity of the data [see, e.g., 121]. We then proceed by considering the nine most popular vendor-items, and further aggregate the remaining ones into a *no-purchase* option, therefore obtaining ten alternatives in total.

The assortment of products shown to the customer at the moment a transaction occurred is not given in the data. Hence, for each transaction $t = 1, \dots, T$ occurring in week w_t at the store s_t, where product j_t was bought, we assume the assortment shown to the customer consists of the set products sold at least once in that store-week combination, i.e., $S_t = \bigcup_{t'=1}^{T} \{j_{t'} : w_{t'} = w_t, z'_t = z_t\}$. At the end of this step we obtain a final list of transactions $\mathcal{T} = \{(S_t, j_t)\}_{t=1}^{T}$.

6.5.2.2 Generalization performance.

Following the experimental setup of Chen and Mišic [37], we separately test the various approach on each of the 29 product categories using 5-folds cross validation. In particular, after considering a product category with transactions data spanning a collection $\mathcal{S} = \{S_1, \dots S_M\}$ of M unique assortments, we partition $\mathcal{S}$ into five (approximately) equally sized collections of assortments $\mathcal{S}_1, \dots \mathcal{S}_5$. For each set $\mathcal{S}_i, i = 1, \dots 5$, let $\mathcal{T}_i$ denote the set of transactions where an assortment belonging to $\mathcal{S}_i$ was shown to the customers, i.e.,$\mathcal{T}_i = \{(S_t, j_t) : S_t \in \mathcal{S}_i\}$. We then run 5 separate experiments where transactions $\mathcal{T}_{train} = \mathcal{T} \setminus \mathcal{T}_i$ are used for training while transactions $\mathcal{T}_i$ are used for testing the predictive accuracy of the algorithms on new assortments.

We measure the predictive accuracy in terms of expected L_1 error, over unseen assortments. We modify equation (6.6) to account for the fact that each test assortment is now associated to a certain number of transactions $T_S = \sum_{t=1}^{T} \mathbb{1}[S_t = S]$. We thus compute the test error as follows:

$$L_1(\boldsymbol{x}, \boldsymbol{v}) = \frac{1}{|\mathcal{T}_{test}|} \sum_{S \in \mathcal{S}_{test}} T_S \sum_{i \in S} \Big| x_{i,S} - v_{i,S} \Big|. \tag{6.7}$$

Table 6.8 Comparison of the various approaches on the IRI Academic Dataset. The reported metric is the average, L_1 error, obtained using 5-folds cross-validation for for each product category.

Product Category	M	T	RB-R	GPT-R	GPT-I	GPT-IC	PCMC	Halo-MNL
Beer	55	380,932	0.2727	0.2729	**0.1272**	0.1619	0.2136	0.1498
Blades	57	92,404	0.0656	0.0565	0.0591	0.0581	0.0732	**0.0429**
Carbonated Beverages	31	721,506	0.1326	0.1284	0.1365	**0.1043**	0.1403	0.1249
Cigarets	68	249,668	0.1396	0.1428	**0.0677**	0.0726	0.0971	0.0765
Coffee	47	372,536	0.1636	0.1681	0.1944	**0.1581**	0.1808	0.1773
Cold Cereal	15	577,236	0.1351	0.1328	0.1021	0.0817	0.0739	**0.0663**
Deodorant	45	271,286	0.0781	0.0791	0.0713	**0.0656**	0.0856	0.0847
Diapers	18	143,055	0.1204	0.1277	0.1305	**0.1120**	0.3627	0.1987
Facial Tissue	43	73,806	0.1185	0.1078	**0.0935**	0.0942	0.1832	0.1173
Frozen Dinners	30	979,936	0.1952	**0.1845**	0.2352	0.2081	0.2213	0.2720
Frozen Pizza	61	292,878	0.1406	0.1427	0.1070	0.1038	0.1184	**0.1027**
Household Cleaners	19	282,981	0.1467	0.1410	**0.1299**	0.1361	0.1313	0.1527
Hotdogs	100	101,624	0.1961	0.1920	0.2064	**0.1733**	0.1983	0.1884
Laundry Detergent	56	238,163	0.1425	**0.1363**	0.1491	0.1479	0.1490	0.1453
Margarine/Butter	18	140,969	0.1750	0.1583	0.1120	0.1264	**0.0975**	0.1371
Mayonnaise	48	97,282	0.0776	**0.0756**	0.0858	0.0818	0.0914	0.0914
Milk	49	240,691	0.1655	0.1618	0.1439	0.1493	0.1359	**0.1243**
Mustard/Ketchup	44	134,800	0.1204	0.1184	**0.0973**	**0.0973**	0.1005	0.1073
Paper Towels	40	82,636	0.1335	**0.1278**	0.1335	**0.1278**	0.1439	0.1654
Peanut Butter	51	108,770	0.1016	0.0991	0.1021	**0.0979**	0.1347	0.1058
Salty Snacks	39	736,148	**0.1300**	0.1381	0.1742	0.1420	0.1378	0.1362
Shampoo	66	290,429	0.1215	0.1184	0.1096	**0.0995**	0.1198	0.1082
Soup	24	905,541	0.1053	**0.1037**	0.1363	0.1199	0.1266	0.1412
Spaghetti/Italian Sauce	38	276,860	0.2017	0.2074	0.2556	0.2049	**0.1857**	0.2100
Sugar Substitutes	64	53,834	0.0800	0.0753	0.0724	0.0723	0.0732	**0.0626**
Toilet Tissue	27	112,788	0.1460	0.1400	0.1284	**0.1113**	0.1477	0.1518
Toothbrush	114	197,676	0.1425	0.1411	0.1145	**0.1128**	0.1313	0.1143
Toothpaste	42	238,271	0.0693	0.0698	**0.0644**	0.0650	0.0742	0.0734
Yogurt	43	499,203	0.1584	0.1703	0.1605	0.1468	**0.1278**	0.1712
Mean	-	-	0.1371	0.1351	0.1276	**0.1184**	0.1399	0.1310
GPT-IC Improvement	-	-	13.7%	12.4%	7.2%	-	15.4%	9.7%
Median	-	-	0.1324	0.1283	0.1200	**0.1086**	0.1271	0.1207
Max	-	-	0.3717	0.3513	0.3438	**0.3110**	0.7175	0.3867
Nb Best	-	-	1	5	6	**11**	3	5

Table 6.8 reports the average test errors of the various approaches on each product category. Columns M and T indicate the total number of unique assortments and transactions, respectively, available for each product category. We further report for each approach the mean, median and maximum test error over *all* experiments. The row "GPT-IC Improvement" reports the percentage improvement of GPT-IC over each approach, while "Nb Best" denotes the number of product categories in which the corresponding approach achieves the

best average predictive accuracy.

We observe that GPT-IC significantly outperforms all other approaches according to all metrics on this dataset. Such results confirm that prioritizing customer behaviors with a relatively small number of relevant, strictly ranked products, may enhance predictive accuracy in practice. Specifically, GPT-IC achieves a 7.2% improvement in predictive accuracy with respect to GPT-I, on average, and a 12.4% improvement with respect to GPT-R, the best RUM baseline. In this context, one may wonder whether the predictive accuracy of GPT-R can be improved by adopting the new customer selection rule of GPT-IC (see Section 6.4.3). We therefore tested such a configuration on the IRI data-set. However, our experiments did not show any improved test errors, confirming that the new dominance rule is indeed helpful only when aiming at estimating irrational customer preference sequences.

Figure 6.5 provides a clearer picture of the improvement one may achieve by going beyond RUM on each product category. In particular, for each category we compare GPT-IC with the *best* performing RUM baseline, namely RB-R and GPT-R, and then report the average percentage improvement in predictive accuracy. Notably, such improvements can be as high as 48% for the category of cigarettes, and close to 40% for the beer and coffee product categories. We also note that for those product categories where the decrease in accuracy is the most significant, such as frozen dinner, laundry and detergent, mayonnaise, salty snacks and soup, none of the irrational baseline we tested was able to outperform the RUM ones. This shows that either such products categories do not contain significant levels of irrationality, or not enough data is available to learn such complex product interactions.

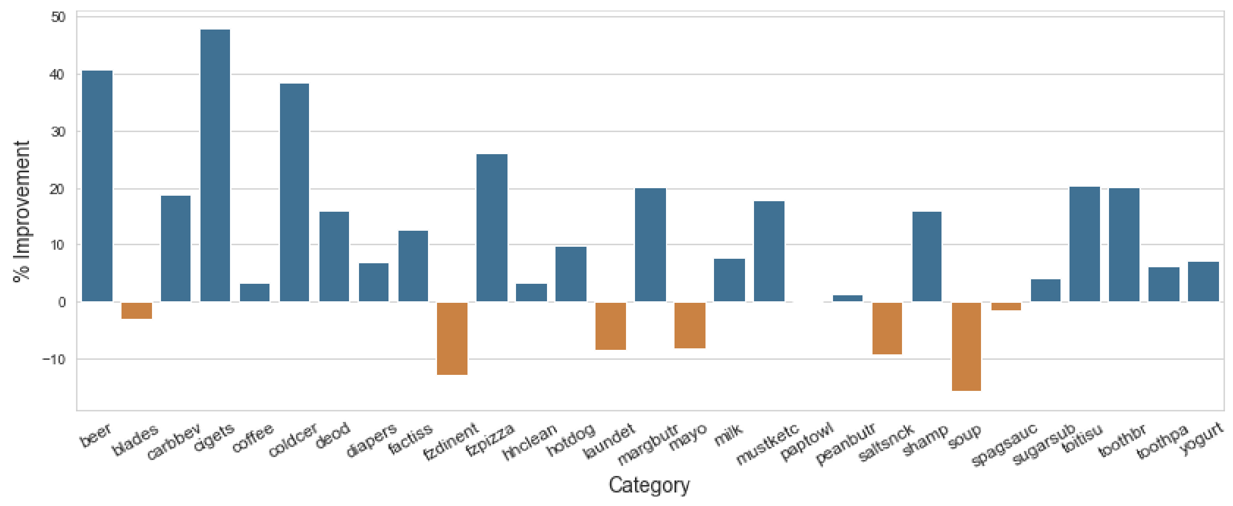


Figure 6.5 Percentage Improvement of GPT-IC over the *best* RUM baseline (between RB-R and GPT-R) in terms of L_1 test error on each Product Category.

Impact of the amount of available data. In this section we investigate how the performance of the various approaches is affected by the amount of data available for training. In particular, Figure 6.6 reports the average test error over all products categories when different amount of transactions are used for training. Specifically, for each of product category, we perform 5-fold cross validations as described above and, for each experiment $i = 1, \ldots, 5$, we select 10%, 25%, 50%, 75% and 100% of the transactions in $\mathcal{T}_{train}$, respectively.

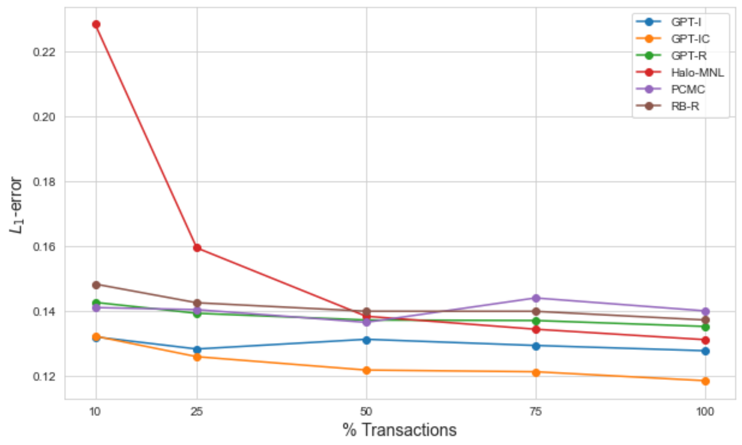


Figure 6.6 L_1 test error of the various approaches, averaged over all product categories, for various amounts of training transactions data.

In line with our results on synthetic instances (see Section 6.5.1), Halo-MNL needs significant amounts of data to be competitive with rank-based approaches, only outperforming GPT-R when 75% or more of the available transactions are used for training. In contrast, the GPT-based approaches seem to be quite data-efficient, with relatively stable performances even for the case where only 10% of the transactions are used for training. Interestingly, the performance of GPT-I slightly deteriorates when moving from 25% to 50% of the transaction being used. This may be due to the fact that, for several product categories, most of the transactions concern a relatively small number of assortments. Hence, cases with only 10% of training transactions tend to exclude "noisy" assortments for which only few transactions (as few as one or two) are available, and which may cause GPT-I to infer spurious product interactions. GPT-IC, on the other hand, does not seem to suffer from such issue, since it prioritizes low-order interactions, which require less data to be learned effectively.

6.6 Conclusion

In this paper, we have proposed a representation for a family of discrete choice models that is sufficiently flexible to capture rational and irrational choice-behavior, together with a computationally- and data-efficient estimation method. The resulting models have been shown to overfit less and generalize well when compared to existing benchmarks.

In particular, we extend the *Generalized Stochastic Preference* choice model introduced by Berbeglia [2] by adapting partially-ranked preference sequences [73], which enables us to estimate the model via column generation. In an effort to prevent additional overfitting induced by the more general irrational preference sequences, and linked to the concept of consideration sets from the marketing literature, we propose a new criterion to select relevant customer types, which are more likely to generalize well in practice.

Our experiments on a popular and extensive set of real-world data has shown that our proposed models have a variety of advantages. First, the use of the new selection rule further improves the generalization accuracy of our irrational choice-model on unseen offer sets by 7.2%, on average. In contrast, using this selection rule in the context of the purely rational choice-model did not result in improvements, confirming that this selection rule is indeed a contribution particular to the irrational case. Second, with respect to the rational baseline models, our irrational model boosts predictive accuracy by 12.4%, on average, and for some categories up to 48%. On (synthetic) rational RUM instances, our irrational choice-models provide stable results, only slightly inferior to the rational baseline models, while the irrational baseline models showed significantly worse performance. Third, our models have shown to be data-efficient, providing a higher predictive accuracy then the benchmarks when little training data is available.

Finally, an appealing feature of our approach is that, within the same framework, it is possible to exploit the explanatory power of (i) partially-ranked preference lists (which can theoretically represent any RUM choice model, see, e.g., Farias et al. [12]) and (ii) irrational behaviors, which can significantly enhance accuracy on irrational instances. Using the new selection rule, our approach is thus capable of providing accurate estimates of product demands on both rational and irrational instances, circumventing the need for effective model selection criteria.

Acknowledgment We would like to acknowledge the Wharton Customer Analytics (WCA) research group and the Majid Al Futtaim (MAF) data & analytics team for the insightful discussions.

CHAPTER 7 GENERAL DISCUSSION

In this chapter, we provide a general overview of our work, and discuss how each of the contributions presented in Chapters 4 to 6 fits into the scope of the thesis. This thesis contributes to the literature on the assortment optimization problem from both the predictive and prescriptive points of view. In particular, we propose scalable algorithms for solving the estimation and decision problems under different choice models. Our goal is to tackle a variety of real-life applications for which different assumptions about the choice behavior of customers may be appropriate.

In Chapter 4, we focus on a variant of the assortment optimization problem under the MNL model. While this model may provide accurate predictions in some choice scenarios, it has well-known limitations (see, e.g., Section 1.1). However, its interpretability is appealing to managers, who may be willing to sacrifice some predictive power in order to gain insights into the buying behavior of customers and "explain" assortment decisions. Such interpretability requirements must be satisfied, in certain situations, in order for the choice-based methodology to be deployed into decision support systems [122, 123]. Furthermore, this model is relatively easy to implement and can be estimated efficiently. Such considerations make the MNL widely adopted in practice, especially when dealing with massive datasets as those encountered in e-commerce use cases, (see, e.g, [27, 124]). Hence, decision problems under such a model are practically relevant. While some of these can be solved efficiently, the literature lacks effective optimization schemes for optimizing assortment decisions under the MNL model in the presence of product costs. These allow to represent strategic scenarios where operational costs are incurred for offering a certain product, and must be taken into account at decision time in order for such costs to be recovered by the firm. We thus filled this gap in the literature by showing that it is possible to identify optimal assortments for this problem in fraction of a second, on average, even on large-scale instances.

For those cases in which customers exhibit complex patterns of substitution, the MNL model may not be able to approximate well enough their buying behavior. If interpretability is not required, one can then resort to more complex models of choice in order to properly fit the given dataset. This may be a difficult process, requiring domain-expert knowledge and extensive model selection by trial and error. Such difficulties stem from the fact that customers may act differently depending on the product or services involved in the decision process. For example, when choosing among products with hedonic features,[1] customers tend to be more variety seeking, often resulting in intransitive preferences [33]. On the opposite, product categories where the impact of brand loyalty is significant may lead to more transitive preferences, which are easier to capture (see, e.g., [19]). Even in the case of brick-and-mortar stores, in which assortment decisions are delivered offline, computational aspects such as the type and amount of available data, or the tractability of the estimation problem, may further drive the decision about which model to apply in a given context. In particular, flexible models can leverage large datasets to alleviate the risk of overfitting, while being able to capture complex choice behaviors. However, effective estimation methods are necessary in order to tackle the computational challenges imposed by these models, especially when dealing with large datasets, so as to benefit from them.

In Chapter 5, we circumvented such difficulties by proposing a fully data-driven approach, which avoids any *a priori* consideration about the market structure by "letting the data speak" instead. In particular, we present a partially-ranked choice model that generalizes popular rank-based choice models and is thus able to approximate any model of the RUM family, to which the MNL belongs. Besides being able to capture *any* pattern of substitution among products, such model allows for a non-parametric estimation procedure that is relatively data efficient, but whose flexibility and computational effectiveness makes it ideal to fully leverage large volumes of sales data. This is a notable result, since alternative estimation methods for general choice models have been shown to be rather computationally costly and prone to overfitting, which make their application more difficult in practice. Moreover, our approach is able to deliver accurate predictions of customer behaviors on new assortments. This, in turn, allows us to consistently identify close-to-optimal assortments, even when relatively little amount of data is available at training time.

The main lesson we learned from our work on partially-ranked preferences in Chapter 5 is that it is possible to avoid the computational burden of estimating fully-ranked preferences while achieving good predictive accuracy. In particular, we empirically observed that customer types forming strict preferences on a small subset of products, while being indifferent to the rest, help explaining sales and generalizing well to new assortments, besides allowing for effective estimation methods. Such observations were the fundamental starting point for our third and final contribution, which we described in Chapter 6. There, we argued that large indifference sets are particularly suited for the estimation of sparse, low-order interactions among products, beside being behaviorally justified. Building on our theoretical intuition,

[1]Are considered *hedonic* those features of a product associated with pleasure, emotions and sensory aspects. Hedonic product categories may include, for example, food, drinks and cigarettes.

we provided an effective estimation method for a choice model which subsumes the models of the RUM family, and is able to capture violations of the regularity assumptions. Such contribution added to a recent stream of literature that focuses on capturing so-called halo effects, whose presence has been shown to be relevant in certain choice scenarios, and must therefore be taken into account when analyzing the buying behavior of customers.

CHAPTER 8 CONCLUSION AND RECOMMENDATIONS

In this thesis, we focused on the problems of discrete choice modeling and assortment optimization, with a special focus on the scalability of the proposed approaches. In the following, we first summarize our contributions, and then highlight possible limitations and future reasearch directions.

8.1 Summary of Works

Our first contribution tackles the problem of optimizing assortment decisions under MNL choice model in the presence of product-specific costs. Although practically relevant, solving this problem to optimality is NP-hard [17], and exact solution methods have been shown to be computationally costly. Hence, most of the literature on the topic focused on approximate solution methods. Building on such works, we proposed a procedure to effectively i) compute relatively tight primal and dual bounds and ii) leverage such bounds to identify the optimal assortment. Our approach outperformed, in terms of computing times, both approximate and exact solutions methods from the literature by several orders of magnitude. Moreover, we have shown how to easily adapt our solution framework to handle side constraints on the assortments, which may arise in practice due to business rules or budget limits.

Incentivized by the always-growing amount of available sales data, more general models of choice have been designed over the years, in order to explain possibly complex buying behaviors. Our second contribution added to this stream of literature by proposing a partially-ranked choice model, which captures the idea of customers being interested in a small fraction of the available products, while being indifferent to the rest. Notably, this model can approximate any choice model belonging to the RUM family, and is therefore able to capture any pattern of substitution among products. We have shown how to efficiently estimate such model from sales data, and how to optimize assortment decisions based on the learned model.

Our last contribution fits into the recent body of work on choice models aimining to escape the limitations of the RUM framework, in order to capture violations of the regularity assumption, commonly referred to as halo effects or product synergies. We have shown how to extend the estimation framework of partially-ranked preferences in order to address the computational and predictive challenges related to the estimation of such models. In order to do so, we proposed principled ways the regularize the use of irrational rank-based behaviors, therefore reducing the risk of overfitting. Our results confirms that accounting for irrational choice behaviors in the estimation process can lead to significant boosts in predictive accuracy, when tested on a real-life grocery sales dataset. Moreover, on such experiments the proposed approach significantly outperformed several baselines from the literature on both RUM and non-RUM choice models in terms of predictive accuracy.

8.2 Limitations and future research directions

The assortment optimization problem with products costs has received relatively little attention in the literature, especially with respect to exact solution methods. Studying this problem in the context of the MNL choice model therefore represented a natural first step on the topic. Beside being a widely adopted model both in industry and academia, the interpretability of the choice probabilities implied by the logit formula, together with their properties, make such model an ideal candidate to investigate about the theoretical and practical tractability of choice-based decision problems encountered in assortment optimization and revenue management. Extending the discussion and methodology outlined in Chapter 4 to more complex choice models or side constraints is an interesting research direction. For example, our method could be swiftly embedded into the solution approach from [52] for (approximately) solving the assortment optimization problem under the MMNL choice model. More generally, our algorithm could be used in all those cases where the assortment optimization problem with product costs appears as subproblem, and may therefore represent a key ingredient in the development of (exact) solution methods for a variety of decision problems.

Our second and third contributions have shown that by assuming that customers may form strict preferences only on a (possibly) small fraction of the available products, one may improve predictive accuracy and obtain more effective estimation procedures. Such observations fit into the more general literature of *consider-then-choose* choice models. This assumes that the decision-making process of an agent can be divided into two steps, where i) she forms a so-called *consideration set*, i.e., determines the subset of products she is interested in, and ii) she chooses an available product among those belonging to her consideration sets; if none of these are available, she leaves without any purchase. Our notion of indifferent sets is complementary to that of consideration set. However, compared to considered-then-choose choice models, where unconsidered products are never chosen by the customer, products in the indifference sets have a non-zero probability of being bought. While we empirically found that indifference sets are important in terms of both convergence of the estimation procedure and generalization performance, we believe our results add to the literature on choice models based on customers' *limited attention* (see, e.g., [87, 118]) and may therefore encourage further developments on the topic.

The predictive accuracy achieved by partially-ranked choice models makes them ideal for identifying high-revenue assortments. To this end, we devoted part of Chapter 4 to show how to optimize assortment decisions based on partially-ranked choice models where customers are assumed to be rational. In particular, we built on the work by Bertsimas and Mišić [56], who proposed a MIP formulation to optimize assortments over fully-ranked preferences (see Problem (5.4) in Chapter 5). Such formulation was shown to provide provably tighter linear relaxations with respect to those in [91] and [55]. Hence, in order to optimize over a partially-ranked choice model, one may be tempted to first generate its equivalent, fully-ranked representation (see Theorem 1 in Chapter 5), and then run the original formulation by Bertsimas and Mišić [56] on such representation. However, while at least $O(n)$ fully-ranked preferences need to be enumerated in order to represent a partially-ranked behavior, with n number of products, the number of necessary fully-ranked preference lists seems to be much larger in practice (see online supplement: Appendix B.1.1), which would make the resulting MIP intractably complex. In this regard, we report in Appendix D.1 an example about the *most concise* fully-ranked representation (i.e., the representation consisting of the smallest number of fully-ranked behaviors) for a given partially-ranked behavior with four items in the indifference set. In order to address this situation, we extended the formulation of Bertsimas and Mišić [56] so as to take into account the presence of indifference sets. Our numerical results confirmed that the proposed approach outperforms fully-ranked choice models in terms of revenue of the generated assortments, while performing well in terms of scalability. Nevertheless, the optimization of assortment decisions in the presence of partially-ranked, *irrational* customer behaviors remains an open research question. In particular, the MIP formulation proposed in Chapter 5 for the rational case cannot be applied as it is in the presence of irrational customer behaviors. The authors' intuition suggests that one may be able to adapt the assortment optimization methodology in [125] to the case of generalized partially-ranked preferences. We consider such investigation as an interesting direction for future research.

We will now briefly pause on the following question: "Are there irrational choice behaviors, other than those described in Chapters 1 and 6, that are empirically robust and should be taken into account when modeling consumers' choice?" One way to circumvent this question is by proposing *universal* choice models that are able to capture *any* discrete choice function and by letting the data speak. We described this stream of literature in Chapter 6. Furthermore, adding to the works therein, Li and Tang [126] and Dogan and Yildiz [127] show that every discrete choice function can be explained in terms of the so-called pro-con choice model or by backward-induction [128], respectively. Both works, however, focus on choice rationalization (i.e., explanation) rather than operational applicability, and do not provide practical methods to estimate customer preferences from data.

In Chapter 6, we proposed a representation of generalized stochastic preferences [2] that is able to capture several widely documented (irrational) choice behaviors (we refer to [103] for an overview). In doing so, we extend the original model from Berbeglia [2] so as to capture the so-called *choice overload* phenomenon, according to which customers may leave with no purchase when faced with an offer set counting too many alternatives. While the robustness and reproducibility of this choice phenomenon is controversial (see, e.g., [129]) we opted once again for a fully data-driven approach, where we allow for the possibility of transaction data being consistent with this type of behavior. However, as mentioned in [2], there do exist choice behaviors that are inconsistent with the generalized stochastic preference choice model. Investigating whether any of such behaviors may be practically relevant is an open research question that, together with a deeper characterization of the GSP choice model, would help understanding whether more generalize models of choice (coming at the cost of higher risk of overfitting) are necessary in certain operational settings.

An important limitation of the non-parametric estimation framework from [30], which we adapted to the case of partially-ranked preferences, stems from its inability to take product features into account in the estimation process. This may impact the application of rank-based choice models in cases where the universe of products is not fixed, i.e., when one needs to predict the market share of a product which has never been sold before. Feature-based approaches, such as the MNL or MMNL, may be able to infer the relative preference among several products based on transactions available for similar ones. More generally, such observations concern the application of rank-based choice models to cases where sales data is sparse, i.e., when only few (or no) transactions are available for most of the *assortment-product* combinations.

In this regard, an interesting line of research concerns the use of assortment optimization methodologies in the context of online Recommender Systems that focus on finding the best subset of products to recommend to customers so as to maximize metrics such as click or conversion rates. This task has been traditionally tackled using independent demand models combined with greedy optimization, due to the tight computational requirements that such systems must satisfy in order to not deteriorate customers' experience. The work of [123] discusses some of the "big data" challenges that may arise in such settings, and how to tackle them in the context of non-parametric rank-based choice models. The work of [27] shows that an MNL model can outperform state-of-the-art Machine Learning models based on the independent demand assumption in terms of revenue coming from recommended products. In the Machine Learning community, such topic is gaining popularity, with more complex models being proposed in order to take into account assortment-dependent effects (such as substitution and halo effects), some of which try to impose inductive biases based on psychological principles of choices. While most of these approaches are purely predictive, some tackle the prescriptive side as well by proposing heuristic methods that optimize the set of recommended products as a whole (see, e.g., [130]). The interplay between the Machine Learning and the Operations Management communities looks thus promising in order to bring behavioral plausibility and optimality guarantees to the former, and scalability and flexibility to the latter.

REFERENCES

[1] J. Feldman and H. Topaloglu, “Bounding optimal expected revenues for assortment optimization under mixtures of multinomial logits,” *Production and Operations Management*, vol. 24, no. 10, pp. 1598–1620, 2015.

[2] G. Berbeglia, “The generalized stochastic preference choice model,” *arXiv preprint arXiv:1803.04244*, 2018.

[3] S. S. Iyengar and M. R. Lepper, “When choice is demotivating: can one desire too much of a good thing?” *Journal of personality and social psychology*, vol. 79, no. 6, pp. 995–1006, dec 2000.

[4] B. Schwartz, *The paradox of choice: Why more is less.* New York, NY, US: HarperCollins, 2004.

[5] G. Vulcano, G. V. Ryzin, and W. Chaar, “Choice-Based Revenue Management : An Empirical Study of Estimation and Optimization,” *Manufacturing & Service Operations Management*, vol. 12, no. 3, pp. 371–392, 2009. [Online]. Available: http://msom.journal.informs.org/cgi/doi/10.1287/msom.1090.0275

[6] R. D. Luce, *Individual choice behavior.* New York: John Wiley & Sons, 1959.

[7] J. Marschak, “Binary choice constraints on random utility indicators,” in *Stanford Symposium on Mathematical Methods in the Social Sciences.* Stanford University Press, 1960.

[8] K. E. Train, *Discrete choice methods with simulation.* Cambridge university press, 2009.

[9] M. E. Ben-Akiva, “Structure of passenger travel demand models.” Ph.D. dissertation, Massachusetts Institute of Technology, 1973.

[10] D. Revelt and K. Train, “Mixed logit with repeated choices: households’ choices of appliance efficiency level,” *Review of economics and statistics*, vol. 80, no. 4, pp. 647–657, 1998.

[11] D. McFadden and K. Train, “Mixed mnl models for discrete response,” *Journal of applied Econometrics*, vol. 15, no. 5, pp. 447–470, 2000.

[12] V. F. Farias, S. Jagabathula, and D. Shah, “A nonparametric approach to modeling choice with limited data,” *Management Science*, vol. 59, no. 2, pp. 305–322, 2013.

[13] I. Simonson and A. Tversky, “Choice in context: Tradeoff contrast and extremeness aversion,” *Journal of marketing research*, vol. 29, no. 3, pp. 281–295, 1992.

[14] J. M. Davis, G. Gallego, and H. Topaloglu, “Assortment Optimization Under Variants of the Nested Logit Model,” *Operations Research*, vol. 62, no. 2, pp. 250–273, 2014. [Online]. Available: http://pubsonline.informs.org/doi/abs/10.1287/opre.2014.1256

[15] R. Y. Maragheh, A. Chronopoulou, and J. M. Davis, “A customer choice model with halo effect,” *arXiv preprint arXiv:1805.01603*, 2018.

[16] R. P. Rooderkerk, H. J. Van Heerde, and T. H. Bijmolt, “Incorporating Context Effects Into a Choice Model,” *Journal of Marketing Research*, vol. 48, no. 4, pp. 767–780, aug 2011.

[17] S. Kunnumkal, P. Rusmevichientong, and H. Topaloglu, “Assortment optimization under multinomial logit model with product costs,” Tech. Rep., 2009.

[18] A. K. Strauss, R. Klein, and C. Steinhardt, “A review of choice-based revenue management: Theory and methods,” *European Journal of Operational Research*, vol. 271, no. 2, pp. 375–387, 2018.

[19] M. C. Cohen, N.-H. Z. Leung, K. Panchamgam, G. Perakis, and A. Smith, “The impact of linear optimization on promotion planning,” *Operations Research*, vol. 65, no. 2, pp. 446–468, 2017.

[20] M. C. Cohen, J. J. Kalas, and G. Perakis, “Promotion optimization for multiple items in supermarkets,” *Management Science*, vol. 67, no. 4, pp. 2340–2364, 2021.

[21] T. W. Gruen, D. S. Corsten, and S. Bharadwaj, “Retail out-of-stocks: A worldwide examination of extent, causes and consumer responses.” Grocery Manufacturers of America., Tech. Rep., 2002.

[22] W. Zinn and P. Liu, “Consumer response to retail stockouts,” *Journal of Business Logistics*, vol. 22, no. 1, pp. 49–71, 2001. [Online]. Available: http://onlinelibrary.wiley.com/doi/10.1002/j.2158-1592.2001.tb00159.x/full

[23] M. Kök, A.G. amd Fisher and R. Vaidyanathan, “Assortment planning: Review of literature and industry practice,” in *Retail Supply Chain Management*, ser. International

Series in Operations Research & Management Science, N. Agrawal and S. Smith, Eds. Springer, 2008, pp. 99–153.

[24] P. M. Guadagni and J. D. Little, "A logit model of brand choice calibrated on scanner data," *Marketing science*, vol. 2, no. 3, pp. 203–238, 1983.

[25] K. Talluri and G. van Ryzin, "Revenue Management Under a General Discrete Choice Model of Consumer Behavior," *Management Science*, vol. 50, no. 1, pp. 15–33, jan 2004. [Online]. Available: http://pubsonline.informs.org/doi/abs/10.1287/mnsc.1030.0147

[26] G. Vulcano, L. N. Stern, G. Van Ryzin, and R. Ratliff, "Estimating Primary Demand for Substitutable Products from Sales Transaction Data," *Operations Research*, vol. 60, no. 2, pp. 313–334, 2012. [Online]. Available: http://dx.doi.org/10.1287/opre.1110.1012

[27] J. Feldman, D. Zhang, X. Liu, and N. Zhang, "Customer choice models versus machine learning: Finding optimal product displays on alibaba," *Available at SSRN 3232059*, 2018.

[28] A. Alptekinoğlu and J. H. Semple, "The Exponomial Choice Model: A New Alternative for Assortment and Price Optimization," *Operations Research*, vol. 64, no. 1, pp. 79–93, 2016. [Online]. Available: http://pubsonline.informs.org/doi/10.1287/opre.2015.1459

[29] J. Blanchet, G. Gallego, and V. Goyal, "A Markov Chain Approximation to Choice Modeling," *Operations Research*, vol. 64, no. 4, pp. 886–905, 2016.

[30] G. van Ryzin and G. Vulcano, "A Market Discovery Algorithm to Estimate a General Class of Nonparametric Choice Models," *Management Science*, vol. 61, no. 2, pp. 281–300, 2015.

[31] D. Bertsimas and V. Mišic, "Data-driven assortment optimization," *Tech. report, Massachusetts Institute of Technology*, 2016.

[32] G. Berbeglia, A. Garassino, and G. Vulcano, "A comparative empirical study of discrete choice models in retail operations," *Available at SSRN 3136816*, 2018.

[33] S. Jagabathula and P. Rusmevichientong, "The limit of rationality in choice modeling: Formulation, computation, and implications," *Management Science*, vol. 65, no. 5, pp. 2196–2215, 2019.

[34] A. Seshadri, A. Peysakhovich, and J. Ugander, "Discovering context effects from raw choice data," in *International Conference on Machine Learning*, 2019, pp. 5660–5669.

[35] J. Kleinberg, S. Mullainathan, and J. Ugander, "Comparison-based choices," in *Proceedings of the 2017 ACM Conference on Economics and Computation*, 2017, pp. 127–144.

[36] S. Ragain and J. Ugander, "Pairwise choice markov chains," in *Advances in Neural Information Processing Systems*, 2016, pp. 3198–3206.

[37] Y. Chen and V. Mišic, "Decision forest: A nonparametric approach to modeling irrational choice," *arXiv preprint arXiv:1904.11532*, 2019.

[38] N. Chen, G. Gallego, and Z. Tang, "The use of binary choice forests to model and estimate discrete choice models," *Available at SSRN 3430886*, 2019.

[39] J. Davis, G. Gallego, and H. Topaloglu, "Assortment planning under the multinomial logit model with totally unimodular constraint structures," *Work in Progress*, 2013.

[40] J. B. Feldman and H. Topaloglu, "Revenue management under the markov chain choice model," *Operations Research*, vol. 65, no. 5, pp. 1322–1342, 2017.

[41] J. J. M. Bront, I. Méndez-Díaz, and G. Vulcano, "A column generation algorithm for choice-based network revenue management," *Operations Research*, vol. 57, no. 3, pp. 769–784, 2009.

[42] P. Rusmevichientong, D. Shmoys, C. Tong, and H. Topaloglu, "Assortment Optimization under the Multinomial Logit Model with Random Choice Parameters," *Production and Operations Management*, vol. 23, no. 11, pp. 2023–2039, nov 2014. [Online]. Available: http://doi.wiley.com/10.1111/poms.12191

[43] P. E. Green and A. M. Krieger, "Models and heuristics for product line selection," *Marketing Science*, vol. 4, no. 1, pp. 1–19, 1985.

[44] D. Honhon, S. Jonnalagedda, and X. A. Pan, "Optimal algorithms for assortment selection under ranking-based consumer choice models," *Manufacturing & Service Operations Management*, vol. 14, no. 2, pp. 279–289, 2012.

[45] G. Gallego and H. Topaloglu, "Constrained assortment optimization for the nested logit model," *Management Science*, vol. 60, no. 10, pp. 2583–2601, 2014.

[46] A. Désir, V. Goyal, D. Segev, and C. Ye, "Constrained assortment optimization under the markov chain–based choice model," *Management Science*, vol. 66, no. 2, pp. 698–721, 2020.

[47] G. Berbeglia and G. Joret, “Assortment optimisation under a general discrete choice model: A tight analysis of revenue-ordered assortments,” *Algorithmica.*, p. Preprint., 2019.

[48] A. Aouad, V. Farias, R. Levi, and D. Segev, “The approximability of assortment optimization under ranking preferences,” *Operations Research*, vol. 66, no. 6, pp. 1661–1669, 2018.

[49] A. Désir, V. Goyal, D. Segev, and C. Ye, “Constrained assortment optimization under the markov chain–based choice model,” *Management Science*, vol. 66, no. 2, pp. 698–721, 2020.

[50] V. Goyal, R. Levi, and D. Segev, “Near-Optimal Algorithms for the Assortment Planning Problem Under Dynamic Substitution and Stochastic Demand,” *Operations Research*, vol. 64, no. 1, pp. 219–235, 2016. [Online]. Available: http://pubsonline.informs.org/doi/10.1287/opre.2015.1450

[51] S. Kunnumkal and V. Martínez-De-Albéniz, “Tractable approximations for assortment planning with product costs,” *Operations Research*, vol. 67, no. 2, pp. 436–452, 2019.

[52] J. B. Feldman and H. Topaloglu, “Capacity constraints across nests in assortment optimization under the nested logit model,” *Operations Research*, vol. 63, no. 4, pp. 812–822, 2015.

[53] I. Méndez-Díaz, J. J. Miranda-Bront, G. Vulcano, and P. Zabala, “A branch-and-cut algorithm for the latent-class logit assortment problem,” *Discrete Applied Mathematics*, vol. 164, no. PART 1, pp. 246–263, 2014.

[54] A. Şen, A. Atamtürk, and P. Kaminsky, “Technical note - A conic integer optimization approach to the constrained assortment problem under the mixed multinomial logit model,” *Operations Research*, vol. 66, no. 4, pp. 994–1003, 2018.

[55] A. Belloni, R. Freund, M. Selove, and D. Simester, “Optimizing product line designs: Efficient methods and comparisons,” *Management Science*, vol. 54, no. 9, pp. 1544–1552, 2008.

[56] D. Bertsimas and V. V. Mišić, “Exact first-choice product line optimization,” *Operations Research*, vol. 67, no. 3, pp. 651–670, 2019.

[57] L. Alfandari, A. Hassanzadeh, and I. Ljubić, “An exact method for assortment optimization under the nested logit model,” *European Journal of Operational Research*, vol. 291, no. 3, pp. 830–845, 2021.

[58] G. Gallego and H. Topaloglu, *Revenue management and pricing analytics.* Springer-Verlag New York, 2019, vol. 279.

[59] A. K. Strauss, R. Klein, and C. Steinhardt, "A review of choice-based revenue management: Theory and methods," *European Journal of Operational Research*, vol. 271, no. 2, pp. 375–387, 2018.

[60] D. McFadden, "Conditional logit analysis of qualitative choice behaviour," in *Frontiers in Econometrics*, P. Zarembka, Ed. New York, NY, USA: Academic Press New York, 1973, pp. 105–142.

[61] J. Feldman and A. Paul, "Relating the approximability of the fixed cost and space constrained assortment problems," *Production and Operations Management*, vol. 28, no. 5, pp. 1238–1255, 2019.

[62] S. Kunnumkal and H. Topaloglu, "A new dynamic programming decomposition method for the network revenue management problem with customer choice behavior," *Production and Operations Management*, vol. 19, no. 5, pp. 575–590, 2010.

[63] P. Rusmevichientong, D. Shmoys, and H. Topaloglu, "Assortment optimization with mixtures of logits," Tech. rep., School of IEOR, Cornell University, Tech. Rep., 2010.

[64] J. Feldman, A. Paul, and H. Topaloglu, "Technical note - Assortment optimization with small consideration sets," *Operations Research*, vol. 67, no. 5, pp. 1283–1299, 2019.

[65] G. Gallego and H. Topaloglu, "Constrained assortment optimization for the nested logit model," *Management Science*, vol. 60, no. 10, pp. 2583–2601, 2014.

[66] N. Liu, Y. Ma, and H. Topaloglu, "Assortment optimization under the multinomial logit model with sequential offerings," *INFORMS Journal on Computing*, vol. 32, no. 3, pp. 835–853, 2020.

[67] D. Honhon, S. Jonnalagedda, and X. A. Pan, "Optimal Algorithms for Assortment Selection Under Ranking-Based Consumer Choice Models," *Manufacturing & Service Operations Management*, vol. 14, no. 2, pp. 279–289, apr 2012.

[68] S. Kunnumkal and H. Topaloglu, "A refined deterministic linear program for the network revenue management problem with customer choice behavior," *Naval Research Logistics (NRL)*, vol. 55, no. 6, pp. 563–580, 2008.

[69] G. P. McCormick, "Computability of global solutions to factorable nonconvex programs: Part I — Convex underestimating problems," *Mathematical Programming*, vol. 10, no. 1, pp. 147–175, 1976.

[70] S. Martello and P. Toth, *Knapsack problems. Algorithms and computer implementations.* John Wiley & Sons Ltd, 1990.

[71] S. Martello and P. Toth, "Upper bounds and algorithms for hard 0-1 knapsack problems," *Operations Research*, vol. 45, no. 5, pp. 768–778, 1997.

[72] ——, "An exact algorithm for the two-constraint 0-1 knapsack problem," *Operations Research*, vol. 51, no. 5, pp. 826–835, 2003.

[73] S. D. Jena, A. Lodi, H. Palmer, and C. Sole, "A partially ranked choice model for large-scale data-driven assortment optimization," *Informs Journal on Optimization*, vol. 2, no. 4, pp. 297–319, 2020.

[74] D. Bertsimas and V. Mišic, "Data-driven assortment optimization," *Tech. report, Massachusetts Institute of Technology*, 2016.

[75] JDA Labs, https://jda.com/innovation/jda-labs, 2017, (Accessed: 2017-09-07).

[76] S. Mahajan and G. J. Van Ryzin, "Retail inventories and consumer choice," in *Quantitative models for supply chain management.* Springer, 1999, pp. 491–551.

[77] A. G. Kök, M. L. Fisher, and R. Vaidyanathan, "Assortment planning: Review of literature and industry practice," in *Retail supply chain management.* Springer, 2008, pp. 99–153.

[78] K. J. Arrow, *Social Choice and Individual Values.* New York, NY, US: John Wiley & Sons, 1951.

[79] M. E. Ben-Akiva, "Structure of Travel Demand Models," Ph.D. dissertation, Massachusetts Institute of Technology, 1973.

[80] S. Jagabathula, "Nonparametric Choice Modeling: Applications to Operations Management," Ph.D. dissertation, Massachusetts Institute of Technology, 2011.

[81] G. Van Ryzin and G. Vulcano, "A Market Discovery Algorithm to Estimate a General Class of Nonparametric Choice Models," *Management Science*, vol. 61, no. 2, pp. 281–300, 2015.

[82] G. Vulcano and G. Van Ryzin, "Technical Note - An expectation-maximization method to estimate a rank-based choice model of demand," *Operations Research*, vol. 65, no. 2, pp. 396–407, 2017.

[83] A. Haensel and G. Koole, "Estimating unconstrained demand rate functions using customer choice sets," *Journal of Revenue and Pricing Management*, vol. 10, no. 5, pp. 438–454, 2011.

[84] N. Ho-Nguyen and F. Kilinc-Karzan, "Dynamic Data-Driven Estimation of Non-Parametric Choice Models," *Working paper, Carnegie Mellon University*, no. Available at arXiv:1702.05702, 2017.

[85] G. Lebanon and Y. Mao, "Non-Parametric Modeling of Partially Ranked Data," *Journal of Machine Learning Research*, vol. 9, pp. 2401—-2429, 2008.

[86] S. Jagabathula and G. Vulcano, "A partial-order-based model to estimate individual preferences using panel data," *Management Science*, vol. 64, no. 4, pp. 1609–1628, 2018.

[87] A. Aouad, V. Farias, and R. Levi, "Assortment optimization under consider-then-choose choice models," *Working paper, Massachusetts Institute of Technology*, no. Available at SSRN 2618823, 2015.

[88] A. Aouad, V. Farias, R. Levi, and D. Segev, "The approximability of assortment optimization under ranking preferences," *Operations Research*, vol. 66, no. 6, pp. 1661–1669, 2018.

[89] S. Jagabathula, "Assortment optimization under general choice," *Working paper, New York University*, no. Available at SSRN 2512831, 2014.

[90] S. Jagabathula and P. Rusmevichientong, "A nonparametric joint assortment and price choice model," *Management Science*, vol. 63, no. 9, pp. 3128–3145, 2016.

[91] R. D. McBride and F. S. Zufryden, "An integer programming approach to the optimal product line selection problem," *Marketing Science*, vol. 7, no. 2, pp. 126–140, 1988.

[92] I. Griva, S. G. Nash, and A. Sofer, *Linear and nonlinear optimization.* Siam, 2009, vol. 108.

[93] J. M. Bront, I. Méndez-Díaz, and G. Vulcano, "A column generation algorithm for choice-based network revenue management," *Operations Research*, vol. 57, no. 3, pp. 769–784, 2009.

[94] H. Palmer, “Large-scale Assortment Optimization,” Master’s thesis, Polytechnique Montréal, 2017.

[95] Z. Huang, “Clustering large data sets with mixed numeric and categorical values,” in *In The First Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 1997, pp. 21–34.

[96] K. T. Talluri and G. J. Van Ryzin, *The Theory and Practice of Revenue Management*. Boston, MA, US: Springer, 2004.

[97] L. L. Thurstone, “A law of comparative judgment.” *Psychological review*, vol. 34, no. 4, p. 273, 1927.

[98] H. Block and J. Marschak, “Random orderings and stochastic theories of response,” Cowles Foundation for Research in Economics, Yale University, Tech. Rep. 66, 1959.

[99] I. Simonson, “Choice based on reasons: The case of attraction and compromise effects,” *Journal of consumer research*, vol. 16, no. 2, pp. 158–174, 1989.

[100] J. Huber, J. W. Payne, and C. Puto, “Adding asymmetrically dominated alternatives: Violations of regularity and the similarity hypothesis,” *Journal of consumer research*, vol. 9, no. 1, pp. 90–98, 1982.

[101] J. Huber and C. Puto, “Market boundaries and product choice: Illustrating attraction and substitution effects,” *Journal of consumer research*, vol. 10, no. 1, pp. 31–44, 1983.

[102] D. Ariely, *Predictably irrational*. New York, NY, US: HarperCollins, 2008.

[103] J. Rieskamp, J. R. Busemeyer, and B. A. Mellers, “Extending the Bounds of Rationality: Evidence and Theories of Preferential Choice,” *Journal of Economic Literature*, vol. 44, no. 3, pp. 631–661, 2006.

[104] J. R. Busemeyer and J. T. Townsend, “Decision field theory: A dynamic-cognitive approach to decision making in an uncertain environment,” *Psychological Review*, vol. 100, no. 3, pp. 432–459, 1993.

[105] R. M. Roe, J. R. Busemeyer, and J. T. Townsend, “Multialternative decision field theory: a dynamic connectionist model of decision making.” *Psychological review*, vol. 108, no. 2, pp. 370–392, 2001.

[106] M. Usher and J. L. McClelland, “Loss aversion and inhibition in dynamical models of multialternative choice.” *Psychological review*, vol. 111, no. 3, p. 757, 2004.

[107] A. Tversky and I. Simonson, “Context-Dependent Preferences,” *Management Science*, vol. 39, no. 10, pp. 1179–1189, 1993.

[108] G. Gallego, R. Ratliff, and S. Shebalov, “A general attraction model and sales-based linear program for network revenue management under customer choice,” *Operations Research*, vol. 63, no. 1, pp. 212–232, 2014.

[109] F. Echenique, K. Saito, and G. Tserenjigmid, “The perception-adjusted luce model,” *Mathematical Social Sciences*, vol. 93, pp. 67–76, 2018.

[110] F. Echenique and K. Saito, “General luce model,” *Economic Theory*, vol. 68, no. 4, pp. 811–826, 2019.

[111] G. Feng, X. Li, and Z. Wang, “On substitutability and complementarity in discrete choice models,” *Operations Research Letters*, vol. 46, no. 1, pp. 141 – 146, 2018.

[112] J. I. Yellott, “The relationship between luce’s choice axiom, thurstone’s theory of comparative judgment, and the double exponential distribution,” *Journal of Mathematical Psychology*, vol. 15, no. 2, pp. 109 – 144, 1977.

[113] T. Osogami and M. Otsuka, “Restricted boltzmann machines modeling human choice,” in *Advances in Neural Information Processing Systems*, 2014, pp. 73–81.

[114] K. Pfannschmidt, P. Gupta, and E. H"ullermeier, “Learning choice functions,” *arXiv preprint arXiv:1901.10860*, 2019.

[115] N. Rosenfeld, K. Oshiba, and Y. Singer, “Predicting choice with set-dependent aggregation,” in *International Conference on Machine Learning*, 2020, pp. 2635–2644.

[116] A. Mottini and R. Acuna-Agost, “Deep choice model using pointer networks for airline itinerary prediction,” in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2017, pp. 1575–1583.

[117] J. R. Hauser, O. Toubia, T. Evgeniou, R. Befurt, and D. Dzyabura, “Disjunctions of conjunctions, cognitive simplicity, and consideration sets,” *Journal of Marketing Research*, vol. 47, no. 3, pp. 485–496, 2010.

[118] S. Jagabathula, D. Mitrofanov, and G. Vulcano, “Inferring consideration sets from sales transaction data,” *NYU Stern School of Business*, 2019.

[119] J. R. Hauser and B. Wernerfelt, “An evaluation cost model of consideration sets,” *Journal of consumer research*, vol. 16, no. 4, pp. 393–408, 1990.

[120] B. J. Bronnenberg, M. W. Kruger, and C. F. Mela, "Database paper—the iri marketing data set," *Marketing science*, vol. 27, no. 4, pp. 745–748, 2008.

[121] B. J. Bronnenberg and C. F. Mela, "Market roll-out and retailer adoption for new brands," *Marketing Science*, vol. 23, no. 4, pp. 500–518, 2004.

[122] J. D. Little, "Models and managers: The concept of a decision calculus," *Management science*, vol. 50, no. 12_supplement, pp. 1841–1853, 2004.

[123] V. F. Farias, S. Jagabathula, and D. Shah, "Building Optimized and Hyperlocal Product Assortments: A Nonparametric Choice Approach," 2017.

[124] A. Aouad, J. Feldman, D. Segev, and D. Zhang, "Click-based mnl: Algorithmic frameworks for modeling click data in assortment optimization," *Available at SSRN 3340620*, 2019.

[125] Y.-C. Chen and V. Mišić, "Assortment optimization under the decision forest model," *Available at SSRN 3812654*, 2021.

[126] J. Li and R. Tang, "Every random choice rule is backwards-induction rationalizable," *Games and Economic Behavior*, vol. 104, pp. 563–567, 2017.

[127] S. Dogan and K. Yildiz, "Every choice function is pro-con rationalizable," *Available at SSRN 3085542*, 2018.

[128] H. W. Kuhn, "11. extensive games and the problem of information," in *Contributions to the Theory of Games (AM-28), Volume II.* Princeton University Press, 2016, pp. 193–216.

[129] Marczyk, Jesse, "Is choice overload a real thing?" https://www.psychologytoday.com/au/blog/pop-psych/201602/is-choice-overload-real-thing, 2016, (Accessed: 2021-11-05).

[130] Y. Gong, Y. Zhu, L. Duan, Q. Liu, Z. Guan, F. Sun, W. Ou, and K. Q. Zhu, "Exact-k recommendation via maximal clique optimization," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2019, pp. 617–626.

[131] A. Domahidi, E. Chu, and S. Boyd, "ECOS: An SOCP solver for embedded systems," in *European Control Conference (ECC)*, 2013, pp. 3071–3076.

[132] S. Diamond and S. Boyd, “CVXPY: A Python-embedded modeling language for convex optimization,” *Journal of Machine Learning Research*, vol. 17, no. 83, pp. 1–5, 2016.

[133] J. Nocedal and S. Wright, *Numerical optimization*, ser. Springer Series in Operations Research. New York, NY, US: Springer, 2006.

APPENDIX A A PARTIALLY-RANKED CHOICE MODEL FOR LARGE-SCALE DATA-DRIVEN ASSORTMENT OPTIMIZATION

A.1 Theoretical Results

In this appendix, we provide theoretical developments and results, such as the definition of a more general choice model, as well as formal proofs.

A.1.1 General partially-ranked customer behaviors

We now define a generalization of the simple partially-ranked customer behavior (see Definition 2). Consider a customer behavior with q lists of preferred products $P^{\ell}(\sigma)$ $(\ell = 1, \dots, q)$ and q indifference sets $I^{\ell}(\sigma)$ as follows.

Definition 6 ***(General partially-ranked customer behavior):*** *A general customer behavior σ has an ordered list of several strictly ranked preference lists $P^1(\sigma), \dots, P^q(\sigma)$ and indifference sets $I^1(\sigma), \dots, I^q(\sigma)$, all of which are mutually exclusive subsets of $\mathcal{N} \cup \{0\}$. We may write such a general customer behavior as $\sigma = (P^1(\sigma), I^1(\sigma), P^2(\sigma), I^2(\sigma), \dots, P^q(\sigma), I^q(\sigma))$.*

As it is the case for simple partially-ranked behaviors, each product from $\mathcal{N}$ can be part of only one strictly ranked list or indifference set. Here, the customer would prefer to buy a preferred product within sequence $P^1(\sigma)$. If none of these products is available, the customer will choose any product with similar characteristics defined in $I^1(\sigma)$ and available in the assortment with uniform probability. If none of those products is available, further lists of preferred products and sets of indifferent products may follow. The position of no-purchase option 0 may result in equivalent notations which are in line with those for simple partially-ranked behaviors mentioned above. In particular, if the no-purchase option 0 is generally available, it can be either in one of the strict preference lists, or in one of the indifference sets. Otherwise, it is implicitly assumed to be at the end of the sequence indicating that no purchase is made.

To illustrate a general partially-ranked behavior, recall the previous example of a simple partially-ranked customer behavior $\sigma = (P(\sigma), I(\sigma)) = (3, 4, 1, \{2, 5, 6\}, 0)$ with one strictly ranked product list and one indifference set. In the case that the customer does not find any of the indifferent products 2, 5 or 6 in the assortment, she may have further strict preferences (e.g., on a different product type), given by another strictly ranked product list, say $(7, 10, 9)$.

If those products are also not available in the assortment, the customer may be indifferent on products 8 and 11. In the absence of those products, she may want to leave the store. This more complex behavior is represented by $\sigma = (3, 4, 1, \{2, 5, 6\}, 7, 10, 9, \{8, 11\}, 0)$, having two strictly ranked products lists and two indifference sets.

While a particular instance of the general partially-ranked behavior could be fit into the taxonomy of [85], the generic general partially-ranked behavior model itself cannot be described by its $\mathfrak{S}$ notation, which assumes that the number, size and order of strictly ranked parts and indifference sets is fixed beforehand.

A.1.2 Verification of Observation 1

We may easily verify Observation 1 by noticing that the probability that the product at k^{th} rank is selected by the customer equals $(1 - r)^{k-1} \cdot r$, where $(1 - r)^{k-1}$ is the probability that none of the $k - 1$ highest ranked products are selected and r is the probability that product k is subsequently selected. The observation follows, as for any $k > 2$, the probability $(1 - r)^{k-1} \cdot r$ reduces as r increases, and the decrease is exponential in k.

A.1.3 Verification of Observation 2

We may verify Observation 2 as follows. There are $|\mathcal{S}| - |P(\sigma_k)|$ products in the indifference set $I(\sigma_k)$. If none of the strictly ranked products is part of the assortment, the contribution to explaining the sales of one of the products in $I(\sigma_k)$ is $\frac{\lambda_k}{|I(\sigma_k)|}$. Further, the probability that none of the strictly ranked products is part of the assortment is, on average, $(1 - r)^{|P(\sigma_k)|}$ (compare Observation 1). The total average contribution of σ_k to a product $i \in I(\sigma_k)$ follows as the product of the previous two terms.

A.1.4 Proof of Lemma 1

Let $F = |I(\sigma_p)|$. We generate $F!$ different fully-ranked preference lists, each containing the items in $P(\sigma_p)$ followed by a different permutation of the elements in $I(\sigma_p)$. For each of those fully-ranked lists, we define an equal probability $(\lambda_p/F!)$. Consider any given assortment S. We now show that the probability of choosing item i in the presence of an assortment S is the same for both choice models. Let $G = |S \cap I(\sigma_p)|$. If $i \notin S$, the sales probability of i is 0 in both cases. If $i \in S$ and $i \in P(\sigma_p)$, then i will be bought with equal probabilities in both cases, since both the partially- and the fully-ranked lists start with the same sequence of products $P(\sigma_p)$.

For the case where $i \in S$ and $i \in I(\sigma)$ we need to show that the probabilities that i is bought given S are equal for the two cases (the partial preference list and the set of fully-ranked lists). For the partially-ranked list, the probability that product i is selected is defined as λ_p/G, i.e., the original probability divided by the number of products that are both in the indifference set and in the assortment. Recall that each of the generated fully-ranked lists has a probability of $(\lambda_p/F!)$. To compute the probability that i is selected, we multiply this probability by the number of fully-ranked lists in which i ranks highest in assortment S. The latter can be obtained in two steps. First, consider only the G items that are both in the assortment and in the indifference set and compute the number of permutations where i is ranked highest. It can be shown (e.g., via induction) that there are $(G-1)!$ permutations in which item i is at first rank (note that i has to be at the first rank to be selected, since all other $G-1$ items are also in the assortment). Then, for each of those $(G-1)!$ permutations, we compute the number of possibilities how to insert the remaining $(F-G)$ items (which are not in S) into the sub-list with G items. This can be computed by dividing the number of all permutations of the F items (those in $I(\sigma_p)$) by the number of permutations of the G already ordered items (which are in the assortment), i.e., $F!/G!$. The final probability that item i is selected given assortment S therefore amounts to $(\lambda_p/F!) \cdot (G-1)! \cdot (F!/G!) = \lambda_p/G$. $\square$

A.1.5 Proof of Theorem 1

In the following, we formally prove Theorem 1. While Lemma 1 refers to simple partially-ranked behaviors as by Definition 2, we here prove valid equivalence for both the simple (Definition 2) and the general (Definition 6) partially-ranked behavior.

To proof the first part of the theorem, consider any preference list $\sigma_c \in \boldsymbol{\sigma}^C$. We may trivially derive an equivalent σ_p from σ_c by defining a corresponding partially-ranked customer behavior σ_p with the following parameters: $q = 1$, $P^1(\sigma_p) = P(\sigma_c)$, $I^1(\sigma) = \varnothing$ and $\lambda_p = \lambda_c$. We may do this for all behaviors in $\boldsymbol{\sigma}^C$ to derive the new choice model. To derive from $(\boldsymbol{\sigma}^P, \boldsymbol{\lambda}^P)$ an equivalent choice model composed of fully-ranked consumer behaviors, consider any $\sigma_p \in \boldsymbol{\sigma}^P$. It can be verified that the transformation of a simple partially-ranked customer behavior with one set of strictly ranked products and one indifference set in Lemma 1 can be generalized to a general partially-ranked customer behavior $(P^1(\sigma), I^1(\sigma), P^2(\sigma), I^2(\sigma), \ldots, P^q(\sigma), I^q(\sigma), 0)$ with several sets of strictly ranked products and several indifference sets. As in Lemma 1, we generate one fully-ranked list for each of the permutations of the products in the indifference sets, resulting in a total of $|I^1(\sigma)|! \cdot |I^2(\sigma)|! \cdot \ldots \cdot |I^q(\sigma)|!$ fully-ranked lists. Consider a product i and define r such that product i is either in $P^r(\sigma)$ or in $I^r(\sigma)$. The equivalence of the probability that i is selected is then proven in the same way as in Lemma 1 using $P^r(\sigma)$ and $I^r(\sigma)$. $\square$

A.1.6 Proof of Theorem 2

We now formally proof Theorem 2. Consider any pair i and j with $i > j$, as well as a customer sequence k. First note that both sets of constraints are defined for the same combinations of i, j, and k, i.e., all i and j within $I(\sigma_k)$. Therefore, if $\sigma^k(i) \neq \sigma^k(j)$, neither of the two sets of constraints are defined, and therefore do not impact the values of y_i^k or y_j^k. If $\sigma^k(i) = \sigma^k(j)$, the equivalence is easiest to show by considering the possible values of binary variables x_i and x_j in a feasible solution. If at least one of the two variables x_i or x_j has value 0, constraints (5.5), as well as constraints (5.7)-(5.8) allow y_i^k and y_j^k to take any value between 0 and 1 (which are coherent with the other constraints in the original model). Their values would therefore be equivalent. If both variables x_i and x_j have value 1, constraints (5.5) with right hand side 0 force y_i^k and y_j^k to take equal values. In the same way, constraints (5.7)-(5.8) will have right hand side 0, and can be combined to $0 \leq y_i^k - y_j^k \leq 0$. It follows that y_i^k and y_j^k have to take equal values. Given that the two set of constraints are active for the same combinations of i, j, and k, which therefore makes them equivalent. $\square$

A.2 Additional Details for the Estimation Algorithms

A.2.1 GPT-based column generation algorithm: Pseudo-code

We here give a detailed description of the GPT-based column generation algorithm. We recall that partially-ranked customer behaviors with indifference subsets are represented by a preference tree, in which explicitly listed nodes refer to strictly ranked products. In the general case (see Definition 6) with several indifferent sets that are strict subsets of $\mathcal{N}$, each indifferent set can be represented by a node that includes several products. However, selecting those sets may be context specific and require input from store managers on how to cluster those products. In the following, we will give details for the simpler case with $\sigma = (P(\sigma), I(\sigma))$ as specified in Definition 3.

Algorithm 1 outlines the complete column generation procedure to build the GPT. At each iteration, the algorithm first explores the sub-behaviors of the behaviors in $\boldsymbol{\sigma}$ (i.e., the sub-behaviors of the sequences that are already in the restricted Master Problem at the current iteration). Note that, even if the direct sub-behaviors of $\boldsymbol{\sigma}$ may not have negative reduced cost (see Appendix A.2.2 for a detailed description on how to compute the reduced costs), behaviors further down the tree may have negative reduced cost. We therefore add to $\boldsymbol{\sigma}$ up to δ sub-behaviors with the smallest reduced costs (even if they are not negative). If none of those sub-behaviors has negative reduced cost, the algorithm iterates up to a defined number of attempts (parameter $\overline{d}$). If, after $\overline{d}$ iterations, no negative reduced cost sub-behaviors have

Algorithm 1: GPT-based column generation algorithm

Input data :

- Sales probability vector $\boldsymbol{v}$, training set of assortments $S_1,\ldots,S_M$.
- Maximum number of column generation iterations $iter^{CG}_{MAX}$.
- Optimality training criteria ϵ^{Tr}_{MIN}.
- Maximum number of sub-behaviors δ that are added at each iteration.
- Maximum number of attempts $\overline{d}$ to find sub-behaviors with negative reduced cost

Output data:

- A set $\boldsymbol{\sigma} = \{\sigma_0, \ldots, \sigma_{K-1}\}$ of K customer behaviors, where $\sigma_k = (P(\sigma_k), I(\sigma_k), 0)$.

```
1  begin
2      Initialize set $\boldsymbol{\sigma} = \{\sigma_0, \sigma_1, \ldots, \sigma_N\}$ defined with $P(\sigma_k) = (k)$ and $I(\sigma_k) = \mathcal{N} \setminus \{k\}$;
3      Set $iter$ to 0;
4      Solve restricted Master Problem (5.1) to obtain $\boldsymbol{\lambda}$, $\boldsymbol{\epsilon}^+$, $\boldsymbol{\epsilon}^-$ and dual values $\boldsymbol{\alpha}$ and
         $\nu$;
5      while ($iter \leq iter^{CG}_{MAX}$) and ($\mathbf{1}^T\boldsymbol{\epsilon}^+ + \mathbf{1}^T\boldsymbol{\epsilon}^- > \epsilon^{Tr}_{MIN}$) do
6          Set $iter \longleftarrow iter + 1$;
7          Set $attempt \longleftarrow 0$;
8          Set $\mathcal{NRC} \longleftarrow False$;
9          while ($\mathcal{NRC} = False$) and ($attempt < \overline{d}$) do
10             Set $attempt \longleftarrow attempt + 1$;
11             for $\forall \sigma_k \in \boldsymbol{\sigma}$ do
12                 if $P(\sigma_k)_{|P(\sigma_k)|} \neq 0$ (i.e., if last element in $P(\sigma_k)$ is not 0) then
13                     compute the reduced costs for all new sub-behaviors of $\sigma_k$;
14                     if $\exists$ sub-behavior with negative reduced cost then
15                         $\mathcal{NRC} \longleftarrow True$;
16             Add to $\boldsymbol{\sigma}$ up to $\delta$ of the generated sub-behaviors with lowest reduced costs;
17         if ($\mathcal{NRC} = False$) then
18             Solve MIP (5.3) to find $\sigma_k$ with smallest reduced cost;
19             if ($\sigma_k$'s reduced cost is negative) then
20                 Add $\sigma_k$ to $\boldsymbol{\sigma}$;
21             else
22                 Return $\boldsymbol{\sigma}$;
23         Set $\boldsymbol{\sigma}$ as new columns to matrix $\boldsymbol{A}$;
24         Solve restricted Master Problem (5.1) to obtain $\boldsymbol{\lambda}$, $\boldsymbol{\epsilon}^+$, $\boldsymbol{\epsilon}^-$ and dual values $\boldsymbol{\alpha}$
             and $\nu$;
25     return $\boldsymbol{\sigma}$
```

been found, MIP (5.3) can be employed to find the most negative reduced cost column. If this cost is positive, optimality has been proven and the algorithm terminates. Otherwise, the column is added and the procedure starts over. Note that our algorithm inherits the typical convergence guarantees from column generation, given that, in the worst case, it will identify the fully-ranked column that has the lowest reduced cost.

The column generation procedure, as outlined in Algorithm 1, explores the sub-behaviors of all $\sigma_k \in \boldsymbol{\sigma}$ (see code line 11). In practice, exploring all nodes may be unnecessarily time consuming. Instead, we may randomly select up to γ behaviors for which the sub-behaviors should be explored. In our computational experiments, we set the maximum number of attempts $\overline{d} = 15$. We also set $\gamma = 10$ and select those behaviors randomly, weighted by their probabilities λ_k.

A.2.2 GPT-based column generation algorithm: Computation of reduced costs

We now describe in detail how the reduced costs of new columns in the Growing Preference Tree are computed. Let us assume that we have computed the reduced costs $rc_{\sigma_c} = rc_{P(\sigma_c)} + rc_{I(\sigma_c)}$ of behavior σ_c, where $rc_{P(\sigma_c)}$ is the part of the reduced costs coming from the strictly ranked set and $rc_{I(\sigma_c)}$ accounts for the part from the indifference set.

Let us also assume that, while having computed rc_{σ_c}, we have also generated a subset $\mathcal{M}_1 \subset \mathcal{M}$ that contains all assortments in which a strictly ranked product from $P(\sigma_c)$ was selected. For each assortment $S_m \notin \mathcal{M}_1$, we also assume to have computed its specific contribution $rc_{I(\sigma_c),m}$, such that $rc_{I(\sigma_c)} = \sum_{m:S_m \notin \mathcal{M}_1} rc_{I(\sigma_c),m}$.

To compute the reduced costs of a sub-behavior of σ_c that additionally ranks item $i \in I(\sigma_c)$, we have to adjust the current reduced cost rc_{σ_c} by distinguishing the following three cases for each of the M assortments:

1. If $S_m \in \mathcal{M}_1$, then adding a strictly ranked item i will not impact the choice (and therefore not the reduced cost).
2. If $S_m \notin \mathcal{M}_1$, then the indifference set was explaining sales in σ_c and contributing to its reduced cost. If $i \in S_m$, it will be selected. One therefore has to add its dual value $\alpha_{i,m}$ and subtract the entire reduced cost $rc_{I(\sigma_c),m}$ stemming from the indifference set.
3. If $S_m \notin \mathcal{M}_1$, but $i \notin S_m$, then the remaining items in the indifference set will contribute to the reduced costs. However, given that i is not in the assortment, it did not contribute to the reduced cost of σ_c, and will not contribute to the reduced cost of the new sub-behavior. Therefore, no changes to the reduced cost have to be made.

Given that cases 1 and 3 do not result in changes, and case 2 is performed in constant time, the total complexity to compute the reduced cost for a sub-behavior is $O(M)$.

APPENDIX B ONLINE SUPPLEMENT - A PARTIALLY-RANKED CHOICE MODEL FOR LARGE-SCALE DATA-DRIVEN ASSORTMENT OPTIMIZATION

B.1 Additional Details and Results for the Choice Model and Estimation Algorithms

In this appendix, we provide additional details and computational results for the proposed choice model and the estimation method.

B.1.1 On the size of equivalent fully-ranked choice models

We here elaborate on the number of fully-ranked preference sequences that may be required to equivalently represent a partially-ranked preference sequence. Let $\sigma_c = (P(\sigma_c), I(\sigma_c))$ be a partially-ranked preference sequence with corresponding probability λ_c. In the following, we will elaborate on lower and upper bounds for the number of fully-ranked preference sequences required to represent the same choice probabilities as given by λ_c. Let us assume, without loss of generality, that $P(\sigma_c) \cup I(\sigma_c) = \mathcal{N} \cup \{0\}$ and define F as the number of elements in the indifference set, thus $F = |I(\sigma_c)|$.

Lower bounds. A trivial lower bound on the minimum number of fully ranked behaviors required to represent σ_c is given by F. To illustrate this, let us assume the contrary, i.e., we would only require $F-1$ fully ranked columns. This means that there exists an option $j \in I(\sigma_c)$ which is never ranked first among the options in the indifference set in the corresponding fully ranked representation. Thus, if we offer an assortment $S = I(\sigma_c)$ we have $P(j|S) = 0$ instead of $\frac{1}{F}$, as it should be. We therefore require at least F fully-ranked sequences.

Exact computation. The previous lower bound might be weak. We therefore empirically investigated how the size of an equivalent fully-ranked choice model may vary as a function of F. We used MIP (B.1), which minimizes the number of fully-ranked sequences required to obtain the same choice probabilities as given by a partially-ranked sequence σ_c. This MIP enumerates, for a given $F = |I(\sigma_c)|$, all $F!$ possible fully-ranked preference sequences and aims at finding a probability mass function $\boldsymbol{\lambda}$ such that the choice probabilities between the two models are equivalent in each of the 2^F possible assortments.

$$\min_{\boldsymbol{y},\boldsymbol{\lambda}} \quad \mathbf{1}^T\boldsymbol{y} \tag{B.1a}$$

$$\text{s.t.} \quad \boldsymbol{A}\boldsymbol{\lambda} = \boldsymbol{a_c}\lambda_c \tag{B.1b}$$

$$\lambda_j \leq y_j \qquad \forall j \in \{1, ..., F!\} \tag{B.1c}$$

$$\mathbf{1}^T\boldsymbol{\lambda} = \lambda_c \tag{B.1d}$$

$$\boldsymbol{\lambda} \geq 0 \tag{B.1e}$$

$$\boldsymbol{y} \in \{0, 1\}^{F!}.$$

To be precise, given the partially ranked behavior σ_c, we compute the column vector $\boldsymbol{a_c}$ as described in Equation (5.2). We further build the binary matrix $\boldsymbol{A}$ as proposed by [74] for all the $F!$ fully-ranked behaviors obtained permuting the elements in the indifference set. We further define binary variables y_j to take value 1 if the fully-ranked sequence j is selected, and 0 otherwise. Problem (B.1) then aims at minimizing the number of selected fully-ranked sequences, such that the choice probabilities given by such behaviors and those given by the partially-ranked behavior are the same (constraints (B.1b)). Note that a fully-ranked column j can have non-zero probability λ_j only if the corresponding y_j variable is selected (constraints (B.1c)), and all the corresponding probabilities needs to sum to λ_c (constraint (B.1d)).

Given the exponential number of assortments to consider, the MIP could be solved only for small values of F. Table B.1 reports the results for F up to 6, showing that the minimum number of columns quickly increases when F grows. However, we can also see that significantly less than $F!$ fully ranked behaviors ("column UB $F!$" may be required to represent a given partially-ranked behavior. In fact, column "UB L" provides an upper bound discussed further below, which is much tighter.

Table B.1 Results on the minimum number of fully-ranked columns required to represent the choices determined by a partially-ranked behavior for any possible assortment

LB F	opt. obj. of MIP (B.1)	UB $F!$	UB L
3	6	6	12
4	12	24	32
5	40	120	80
6	60	720	192

Upper bound. Let us consider the feasibility problem obtained from the previously presented MIP (B.1) by removing variables $\boldsymbol{y}$ and constraints (B.1c), and by using a constant objective function. Let also $\mathcal{P}$ be the power set of $\{1, ..., F\}$. The resulting coefficients matrix would have $L = \sum_{S \in \mathcal{P}} |S|$ rows, and $F!$ columns. Any basis solution of the feasibility problem will have at most L linearly independent columns, which is therefore an upper bound on the number of different solutions that may exist. We do not have a closed form solution for L, but we can bound it via $L < 2^F \cdot F < F!$, if F is sufficiently large. We can therefore conclude that, in general, we need less than $F!$ columns to represent a partially-ranked behavior using only fully-ranked ones. The values of L for F up to 6 can be found in Table B.1.

B.1.2 CG-LS: Tuning with different neighborhoods

We here report on the computational results for different implementations of the local-search based column generation algorithm (CG-LS) proposed by [74]. The experiments reported in Table 5.1 show that CG-GPT and CG-LS generally need a similar number of behaviors K to reach convergence. However, the time required by CG-LS to find each of these behaviors becomes prohibitive when the number of products N increases. This is due to the fact that the local search, as proposed by [74], swaps the rank of any two elements in a preference list. Thus, for any permutation, $\frac{N(N-1)}{2} \in O(N^2)$ swaps have to be performed. One may wonder whether a faster implementation of this local search might improve the scalability of the approach. We therefore tested the following local-search variants for the CG-LS:

- *Small neighborhood*: swap only the rank of consecutive products, i.e., of products with ranks r and $r + 1$. This is done up to a fixed number of swaps (we use 10 swaps for each permutation).

- $O(N^2)$ *neighborhood*: we optimize the original local search using the same k-deletion principle of [33]. In particular, given a collection of assortments $\mathcal{M}$, at most $\bar{n} = \max_{S \in \mathcal{M}} N - |S|$ products are missing. This means that, in order to determine a choice in any of the assortments in $\mathcal{M}$, only $\bar{n} + 1$ products need to be ranked. We can thus avoid swapping elements (i, j) if $\sigma(i) > \bar{n} + 1$ and $\sigma(j) > \bar{n} + 1$.

- $O(N)$ *neighborhood*: this neighborhood is inspired by the number of neighbors explored by CG-GPT at every iteration. In fact, CG-GPT samples γ behaviors and then explores $|I(\sigma_c)|$ sub-behaviors for each of them, resulting in a total of $O(N)$ neighbors. We adapt the CG-LS local search to sample $O(N)$ different couples of products whose rank is swapped.

The variants above have been tested both with an initialization consisting of one random preference list (as done for Table 5.1) and with the Independent Demand initialization technique (see [81]), which initializes the algorithm with one *singleton* preference list for each product (singleton lists are preference lists where the no-purchase option 0 is ranked second).

Table B.2 Comparison of choice models obtained with variants of CG-LS

	Small neighborhood											
	Independent demand init						single list init					
N	Train error	Test error	# iter	time (min)	K	# inst. unsolved	Train error	Test error	# iter	time (min)	K	# inst. unsolved
30	0.39	4.03	1,155.1	1.0	233.4	0	0.38	4.00	1,127.3	1.1	235.8	0
50	0.40	4.31	2,434.1	4.5	428.8	0	0.39	4.30	2,395.3	3.9	427.4	0
100	0.40	4.64	5,252.9	29.8	878.4	0	0.40	4.67	5,115.9	30.9	872.9	0
250	0.40	4.57	8,965.0	519.0	2,193.0	8	0.40	4.59	9,001.5	654.3	2,190.5	8
all (avg)	0.40	4.39	4,451.8	138.6	933.4	28	0.39	4.39	4,410.0	172.5	931.7	28
	$O(N^2)$neighborhood											
	Independent demand init						single list init					
N	Train error	Test error	# iter	time (min)	K	# inst. unsolved	Train error	Test error	# iter	time (min)	K	# inst. unsolved
30	0.39	3.93	393.0	0.7	191.9	0	0.40	3.90	365.9	0.7	193.3	0
50	0.40	4.36	741.2	4.0	360.4	0	0.40	4.31	665.2	3.2	343.1	0
100	0.40	4.40	1,361.4	30.3	719.7	0	0.40	4.49	1,274.0	30.8	697.1	0
250	0.40	4.43	2,557.0	432.3	1,686.4	3	0.40	4.49	2,310.5	428.4	1,693.2	4
all (avg)	0.40	4.28	1,263.2	116.8	739.6	23	0.40	4.30	1,153.9	115.8	731.7	24
	$O(N)$neighborhood											
	Independent demand init						single list init					
N	Train error	Test error	# iter	time (min)	K	# inst. unsolved	Train error	Test error	# iter	time (min)	K	# inst. unsolved
30	0.37	3.92	283.7	0.7	196.1	0	0.39	3.82	259.4	0.7	192.9	0
50	0.39	4.28	582.6	4.3	354.0	0	0.40	4.26	555.6	3.7	353.6	0
100	0.39	4.55	1,160.9	34.0	727.0	0	0.40	4.53	1,076.0	28.9	703.4	0
250	0.40	4.57	2,461.7	405.1	1,763.7	4	0.40	4.33	2,188.8	310.6	1,703.2	5
all (avg)	0.39	4.33	1,122.2	111.0	760.2	24	0.40	4.23	1,020.0	86.0	738.3	25

The results presented in Table B.2 show that none of the above described variants is able to deal with instances with more than 250 products (again, we omitted lines of N for which no problem instance was trained). Further, the final numbers of behaviors tend to be slightly larger than those obtained with the local search of [74] (see Table 5.1 and also Table B.7 further below).

B.1.3 Comparison of k-deletion and CG-GPT for the exact case

The k-deletion technique has been proposed in [33] for dealing with cases where only $k-1$ products are missing from the assortment and $k << N$, which is the typical case for stores with high service level and frequent stock-outs and replenishment. Even though such cases may not be common in practice, we here compare the performance of the k-deletion and CG-GPT for cases where k is small. Table B.3 reports training and test errors for both approaches, the average time required to estimate the choice models and the average number of customer behaviors obtained in each case. Given that sales data were generated according to a MMNL model, which belongs to the RUM class of choice models, k-deletion can achieve a 0 training error in all cases. This, however, seems to lead to overfitting, as shown by the test errors, which are higher than those obtained by the CG-GPT (except for very small instances with 10 products).

Table B.3 properties of choice models generated by CG-GPT (with $\epsilon_0 = 0.01$) and k-deletion when the number of missing products $k-1 << N$ ($M = 20$)

		CG-GPT				k-deletion-exact				
# prod. missing	N	Train error	Test error	time (min)	K	Train error	Test error	time (min)	K	# inst. unsolved
2	10	0.33	0.71	<1	41.5	0.00	0.45	<1	112.0	0
2	20	0.28	0.79	<1	54.9	0.00	0.96	<1	318.5	0
2	30	0.25	0.94	<1	61.7	0.00	1.14	<1	519.3	0
2	50	0.21	0.52	<1	82.0	0.00	0.80	<1	909.2	0
2	70	0.26	0.54	<1	92.8	0.00	0.71	<1	1,302.8	0
2	100	0.26	0.33	<1	109.2	0.00	0.51	2.4	1,894.7	0
2	250	0.14	0.10	<1	250.0	0.00	0.25	52.1	4,783.4	0
2	all (avg)	0.25	0.56	<1	98.9	0.00	0.69	7.9	1,405.7	0
3	10	0.30	1.12	<1	50.5	0.00	0.88	<1	110.2	0
3	20	0.30	0.88	<1	71.3	0.00	1.01	<1	317.0	0
3	30	0.33	1.15	<1	71.2	0.00	1.46	1.8	514.6	0
3	50	0.23	0.76	<1	91.9	0.00	1.21	16.0	909.5	0
3	70	0.28	0.65	<1	99.2	0.00	1.09	63.8	1,301.3	4
3	100	0.26	0.42	<1	119.2	-	-	-	-	10
3	250	0.17	0.14	<1	251.9	-	-	-	-	10
3	all (avg)	0.27	0.73	<1	107.9	0.00	1.13	16.4	630.5	24

One may argue that the reason for overfitting is having trained the model to a 0 training error. We therefore also implemented a "regularized" variant. Here, instead of minimizing the ℓ_1 error, we minimize a constant function adding the constraint that $||\boldsymbol{A\lambda} - \boldsymbol{v}|| \leq \epsilon_0 \cdot 2M$. This allows us to accept any solution with the same accuracy threshold as used by the CG-GPT. However, the results suggested that test errors obtained with this variant are generally worse. We therefore do not explicitly report on those experiments.

B.1.4 Varying the Ground Truth: Impact on performance and choice model characteristics

From a managerial perspective one may be interested in a choice model that facilitates insights into the market segmentation and preferences. In this regard, customer behaviors with only few strictly ranked products may give direct insights on which products are the most relevant ones to explain sales. In line with such observation, a practically desired property is to generate strictly ranked preference lists that are as short as possible. We denote this property as *concision*. One may suspect that either the number of products that have high utilities for the customers, or the number of products in the assortment will have an impact on the number of strictly ranked products in the final preference sequences. However, as will be shown next, our computational results (see Tables B.4 and B.5) suggest that the number of strictly ranked products remains rather small (and never exceeds 14 products), therefore supporting Observations 1 and 2 in Section 5.3.

Impact of number of high-utility products in ground truth. Recall that in the experiments above, the ground-truth model to generate the problem instances contains, for each customer class, four products with high utilities. We now explore how the number of products with high utilities in the underlying ground-truth model impacts the final choice model. Table B.4 summarizes the results for $\epsilon_0 = 0.01$, based on ground-truth models that assume that each customer class has exactly 1, 4, 10 and 20 products with high utilities. The last two columns reveal information about the explanatory power of the indifference sets, i.e., the percentage of sales that are explained by indifference sets. The empirical percentage (column "empir") for a given choice model can be computed as $\sum_{k \in \boldsymbol{\sigma}} \lambda_k \cdot \frac{numInd_k}{M}$, where $numInd_k$ is the number of (k, m) tuples in which at least one product from an indifference set has a value greater than 0. In words, it is the weighted ratio between the number of assortments in which a product from the indifference set has been sold and the total number of sales (which equals the total number of assortments M, if the selection of 0 is considered a sale). On the other hand, column "theor" is linked to the theoretical approximation of the indifference percentage in Observation 2. In this observation, the empirical percentage is computed for an average assortment, explicitly using the different λ_k values for each preference list k. Since we are not dealing with average assortments, taking all λ_k into account does not make the result more informative for our case of specific assortments. Column "theor" therefore reports the value given by the simplified formula $(1 - r)^{avgRanked}$, where

Table B.4 Properties of choice models generated by CG-GPT (average values over 10 random instances) with different numbers of products with high utilities in ground-truth model ($\epsilon_0 = 0.01$).

		Train	Test	#	time		# strictly		% explained by		# inst.
	N	error	error	iter	(min)	K	ranked products		indifference sets		unsolved
							avg	max	empir	theor	
$nhu = 1$	30	0.36	1.18	5.1	<1	124.2	2.08	4	26.38	23.79	0
	50	0.38	1.28	9.1	<1	210.9	2.11	4	25.90	23.40	0
	100	0.39	1.26	15.9	<1	390.9	1.95	4	34.60	25.98	0
	250	0.40	1.16	39.1	3.4	965.4	1.86	4	37.75	27.57	0
	500	0.40	0.91	55.6	17.3	1,543.7	1.73	3	44.24	30.28	0
	1,000	0.39	0.68	58.0	77.3	2,055.1	1.44	3	46.67	37.60	0
	avg/max all	0.39	1.08	30.5	16.4	881.7	1.86	4	35.92	28.11	0
$nhu = 4$	30	0.37	2.88	15.9	<1	170.7	3.20	6	7.56	11.03	0
	50	0.38	2.90	31.1	<1	310.5	3.18	6	7.86	11.23	0
	100	0.39	2.72	46.9	<1	619.0	2.77	6	11.75	14.86	0
	250	0.39	2.37	90.6	16.6	1,474.1	2.40	5	18.96	19.17	0
	500	0.40	1.89	116.5	76.1	2,437.1	2.02	4	29.12	24.69	0
	1,000	0.40	1.30	159.7	306.0	3,876.7	1.83	4	38.99	28.11	0
	avg/max all	0.39	2.34	76.8	66.6	1,481.4	2.57	6	19.04	18.18	0
$nhu = 10$	30	0.38	4.30	22.7	<1	176.1	3.42	7	4.01	9.49	0
	50	0.39	4.64	49.2	<1	315.5	3.40	6	3.60	9.53	0
	100	0.39	4.89	81.0	1.7	660.2	3.13	7	4.91	11.45	0
	250	0.40	4.49	154.5	38.6	1,731.9	2.79	6	8.95	14.52	0
	500	0.40	3.47	211.3	274.7	3,255.6	2.39	5	16.64	19.07	0
	1,000	-		-	-	-	-	-	-	-	10
	avg/max all	0.39	4.36	103.7	63.0	1,227.9	3.03	7	7.62	12.81	10
$nhu = 20$	30	0.37	2.53	10.3	<1	176.3	2.44	4	17.40	18.53	0
	50	0.38	3.78	26.4	<1	298.8	2.95	5	7.92	13.14	0
	100	0.39	4.86	67.7	1.0	572.8	3.08	5	4.34	11.85	0
	250	0.40	5.09	158.5	29.5	1,615.5	2.89	6	6.65	13.48	0
	500	0.40	4.53	232.4	241.5	3,199.2	2.55	5	11.33	17.09	0
	1,000	-	-	-	-	-	-	-	-	-	10
	avg/max all	0.39	4.16	99.1	54.4	1,172.5	2.78	6	7.56	14.82	10

avgRanked is the average value reported in column "# strictly ranked products avg" of the same line (with $r = 0.5$). As the number of products with high utilities (nhu) increases, the algorithm requires more iterations to find a choice model that fits the transaction data accurately. However, the number of preference lists with non-negative probabilities remains similar in all cases. The number of strictly ranked products also remains surprisingly stable, indicating that a final accurate choice model is not more complex, but only more difficult to find. Finally, one observes that such more refined choice models also reduce the percentage of sales transactions that are explained by the products in the indifference sets. Finally, it is notable that the theoretical estimation of this percentage is close to that practically computed, which confirms our theoretical findings in Observation 2.

Table B.5 Properties of choice models generated by CG-GPT (average values over 10 random instances) with different assortment densities r ($\epsilon_0 = 0.01$).

r	N	Train. error	Test error	# iter	time (min)	K	# strictly ranked products avg	# strictly ranked products max	# inst. unsolved
0.1	30	0.28	4.65	6.4	<1	34.1	4.95	8	0
0.1	50	0.36	6.02	12.2	<1	63.3	6.16	13	0
0.1	100	0.37	5.62	24.9	<1	132.7	6.01	14	0
0.1	250	0.39	6.08	81.7	2.4	372.5	6.57	13	0
0.1	500	0.40	4.68	171.6	21.5	751.7	5.92	13	0
0.1	1,000	0.40	4.51	380.5	300.6	1,552.1	5.34	12	0
0.1	avg/max all	0.37	5.26	112.9	54.1	484.4	5.83	14	0
0.3	30	0.37	4.13	12.2	<1	110.1	4.36	8	0
0.3	50	0.39	5.07	27.8	<1	204.4	4.42	9	0
0.3	100	0.39	4.64	47.9	<1	443.1	3.98	8	0
0.3	250	0.39	4.03	105.6	14.3	1,117.9	3.63	7	0
0.3	500	0.40	3.18	150.3	78.5	2,073.6	3.09	7	0
0.3	1,000	0.40	2.24	236.9	510.6	3,948.0	2.68	5	3
0.3	avg/max all	0.39	3.91	96.8	100.7	1,316.2	3.69	9	3
0.5	30	0.37	2.88	15.9	<1	170.7	3.20	6	0
0.5	50	0.38	2.90	31.1	<1	310.5	3.18	6	0
0.5	100	0.39	2.72	46.9	<1	619.0	2.77	6	0
0.5	250	0.39	2.37	90.6	16.6	1,474.1	2.40	5	0
0.5	500	0.40	1.89	116.5	76.1	2,437.1	2.02	4	0
0.5	1,000	0.40	1.30	159.7	306.0	3,876.7	1.83	4	0
0.5	avg/max all	0.39	2.34	76.8	66.6	1,481.4	2.57	6	0

Impact of assortment density r. [67] and [33] note that when choosing from an offer set of size $N - \overline{n}$ (where $\overline{n}$ is the number of products that are not present in the assortment), only the first $\overline{n} + 1$ ranks matter (while the remainder will not have any explanatory power at all). While this a theoretical bound, in practice, we argue that the number of relevant ranks may actually be much smaller. The numerical studies reported above indicate that for the CG-GPT, one requires to strictly rank a rather small number of products, while the indifference set can significantly contribute to explain sales. We conducted further experiments to explore the properties of choice models generated by the CG-GPT for different assortment densities r (defined in Observation 1 as the ratio between the number of products in the assortment $|S|$ and the total number of products N, i.e., $r = (N - \overline{n})/N$). The results, summarized in Table B.5, show that low assortment densities r result in slightly higher maximum and average numbers of strictly ranked products. However, they are always significantly smaller than $\overline{n} + 1$, given the explanatory power of the indifference sets. A look at the test errors further suggests that prediction becomes harder for small densities r, most likely because the number of historical observations for each of the products is smaller.

B.1.5 Varying the training accuracy threshold: Impact on performance and choice model characteristics

We now explore how the training accuracy threshold ϵ_0 impacts the prediction accuracy and the produced choice models. Table B.6 shows several properties for the choice models generated by CG-GPT for different training accuracy values $\epsilon_0 \in \{0.1, 0.01, 0.001\}$ and problem sizes N. The results are averaged over 10 random instances and include the average size K of the choice models and the number of strictly ranked products (average and maximum number).

The results indicate that, as the training is more accurate and ϵ_0 is decreased, the number of required GPT iterations and the size of the final choice model increase. The number of strictly ranked items also slightly increases. However, it generally remains quite low and never exceeds more than 7 strictly ranked products in any of the generated preference lists, which is only a fraction of the total number of products (i.e., up to 1000). The proportion of sales explained by products in the indifference set steadily decreases as the training accuracy is increased (i.e., ϵ_0 is decreased). This illustrates the high explanatory power of the first few ranked products if the choice model is well chosen. For example, with $\epsilon_0 = 0.001$ and $N = 250$ products, all preference lists contain 6 or less strictly ranked products, which explain 86.88 % of the sales transactions, while only 13.12 % of the transactions are explained by the remaining 244 products that are not strictly ranked. While classical approaches using fully-ranked preference lists will always contain N strictly ranked products, the proposed approach based on partially-ranked preference lists allows store managers to gain valuable insights from a small list of products that have a fairly high explanatory power. As it has been the case before, the theoretical estimation of this percentage is quite close to that practically computed, confirming our theoretical findings in Observation 2. We finally also point out a key difference to the k-deletion heuristic previously discussed. The k-deletion heuristic, ranking exactly 3 products, obtained an average test error of 6.12 (see Table 5.2). The CG-GPT also strictly ranks not more than 3 products (when using $\epsilon_0 = 0.1$), but obtains a much lower average test error of 3.70. This illustrates the importance of the complementary indifference set to improve predictive accuracy.

For the sake of completeness, we report in Table B.7 the results obtained for CG-LS with

Table B.6 Properties of choice models generated by CG-GPT (average values over 10 random instances) with different training error thresholds ϵ_0.

ϵ_0	N	Train error	Test error	# iter	time (min)	K	# strictly ranked products avg	# strictly ranked products max	% explained by indifference sets empir	% explained by indifference sets theor	# inst. unsolved
0.1	30	3.15	4.90	1.6	<1	32.5	1.90	3	25.73	27.05	0
0.1	50	3.22	5.17	2.1	<1	48.2	1.79	3	26.50	29.30	0
0.1	100	3.35	4.67	1.5	<1	71.9	1.40	3	31.76	38.15	0
0.1	250	2.83	3.60	1.1	<1	171.1	1.13	2	40.20	45.87	0
0.1	500	1.98	2.41	1.0	<1	389.2	1.05	2	46.94	48.30	0
0.1	1,000	1.29	1.48	1.0	1.5	854.6	1.02	2	49.97	49.28	0
0.1	avg/max all	2.63	3.70	1.4	< 1	261.3	1.38	3	36.85	39.66	0
0.01	30	0.37	2.88	15.9	<1	170.7	3.20	6	7.56	11.03	0
0.01	50	0.38	2.90	31.1	<1	310.5	3.18	6	7.86	11.23	0
0.01	100	0.39	2.72	46.9	<1	619.0	2.77	6	11.75	14.86	0
0.01	250	0.39	2.37	90.6	16.6	1,474.1	2.40	5	18.96	19.17	0
0.01	500	0.40	1.89	116.5	76.1	2,437.1	2.02	4	29.12	24.69	0
0.01	1,000	0.40	1.30	159.7	306.0	3,876.7	1.83	4	38.99	28.11	0
0.01	avg/max all	0.39	2.34	76.8	66.6	1,481.4	2.57	6	19.04	18.18	0
0.001	30	0.03	2.69	51.4	<1	263.8	3.78	7	3.49	7.43	0
0.001	50	0.04	2.77	78.4	<1	451.6	3.65	7	4.20	8.17	0
0.001	100	0.04	2.64	98.8	4.1	926.1	3.17	7	7.52	11.26	0
0.001	250	0.04	2.39	179.3	81.4	2,323.2	2.73	6	13.12	15.17	0
0.001	500	0.04	1.87	261.2	660.8	4,566.3	2.28	5	21.92	20.57	4
0.001	1,000	-	-	-	-	-	-	-	-	-	10
0.001	avg/max all	0.04	2.62	102.0	149.4	991.2	3.33	7	7.08	10.51	14

different training accuracy threshold. We have omitted lines for instance sizes where none of the 10 problems have been solved by CG-LS. The results indicate that the CG-LS is able to estimate larger problem instances when using a higher accuracy threshold of 0.1 (instead of 0.01). However, the test errors significantly increase, which is not a desirable trade-off.

B.1.6 General convex loss function: maximum likelihood estimator.

As discussed in Section 5.3.2, the CG-GPT method can be framed into the column-generation approach proposed by [81]. Even though we have shown how to estimate the choice probabilities minimizing the absolute ℓ_1 error, our approach can be used minimizing any other convex loss function (since strong duality still holds to prove optimality; see e.g., [92] Theorem 14.37). We may therefore attempt to estimate the choice probabilities by maximizing the likelihood probability as proposed by [81] or [86]. In line with those works, we have implemented an objective function that minimizes the sum of Kullback divergences[1] between

[1] [86] actually propose to minimize a weighted sum of Kullback divergences, where the weight is the number of customer exposed to a given assortment. For our generated data, however, we can assume that all the

Table B.7 Learning choice models with CG-LS (averaged over 10 random instances) with different training error thresholds $\epsilon_0 (M = 20)$.

		#		Train	Test	time	# inst.
ϵ_0	N	iter	K	err	err	(sec)	unsolved
0.1	30	48.1	43.9	3.94	6.99	12.2	0
0.1	50	91.2	77.5	3.95	7.24	65.8	0
0.1	100	186.2	164.7	3.94	7.53	550.9	0
0.1	250	431.4	401.0	3.99	7.24	8,734.3	0
0.1	all (avg)	189.2	171.8	4.0	7.2	2,340.8	20
0.01	30	309.0	177.5	0.39	3.74	88.7	0
0.01	50	619.0	337.3	0.40	4.22	501.1	0
0.01	100	1,183.1	667.7	0.40	4.41	4,579.5	0
0.01	all (avg)	703.7	394.2	0.4	4.1	1,723.1	30

the estimated choice probabilities $\hat{\boldsymbol{p}}$ and observed probabilities $\boldsymbol{v}$ for all assortments:

$$loss(\boldsymbol{v}, \hat{\boldsymbol{p}}) = -\sum_{(i,m)} v_{i,m}\ log\ \frac{\hat{p}_{i,m}}{v_{i,m}} \tag{B.2}$$

While [33] use the Frank-Wolfe algorithm to estimate the probabilities within the restricted Master problem, we use an off-the-shelf Non-linear Programming (NLP) solver. Specifically, we use the ECOS [131] NLP solver with python library `cvxpy` [132], stopping the training when $\max_{i,m} |\hat{p}^k_{i,m} - \hat{p}^{k+1}_{i,m}|$ is smaller or equal to 0.001 at two consecutive iterations k and $k+1$.

In order to compare the performance when using the different objective functions and training criteria, Table B.8 reports both the ℓ_1 error and the root mean squared error[2] (RMSE) for both training and test data in each of the experiments. We also report the average computing times and the average number of behaviors of the final choice models for each problem size. Optimizing the KL divergence seems to be a good trade-off both in terms of goodness of fit and computing time when compared to the training with ℓ_1-norm objective function and different accuracy thresholds. Moreover, despite being less sparse (i.e., having larger K) on smaller instances, it converges with less columns on bigger instances, which is important from an optimization point of view. It is worth noting, however, that the used NLP solver encountered convergence problems on medium and large instances, for which additional care

assortments have been shown to the same number of customers.

[2]The RMSE is defined as $\sqrt{\frac{\sum_{(i,m)} (\hat{p}_{i,m} - v_{i,m})^2}{Q}}$ where $\hat{p}_{i,m}$ are the estimated purchase probabilities of object i in assortment S_m, where $v_{i,m}$ are the observed probabilities, while Q is the number of (i, m) observations.

Table B.8 Learning using CG-GPT with different objective functions (averaged over 10 random instances).

obj	ϵ_0	N	ℓ_1 Train error	ℓ_1 Test error	RMSE Train error	RMSE Test error	# iter	time (min)	K	#inst. unsolved
ℓ_1	0.1	30	3.15	4.90	0.0218	0.0308	1.6	< 1	32.5	0
ℓ_1	0.1	50	3.22	5.17	0.0155	0.0220	2.1	< 1	48.2	0
ℓ_1	0.1	100	3.35	4.67	0.0091	0.0122	1.5	< 1	71.9	0
ℓ_1	0.1	250	2.83	3.60	0.0042	0.0052	1.1	< 1	171.1	0
ℓ_1	0.1	500	1.98	2.41	0.0016	0.0020	1.0	< 1	389.2	0
ℓ_1	0.1	1,000	1.29	1.48	0.0006	0.0007	1.0	1.5	854.6	0
		all (avg)	2.63	3.70	0.0088	0.0122	1.4	< 1	261.3	0
ℓ_1	0.01	30	0.37	2.88	0.0042	0.0194	15.9	< 1	170.7	0
ℓ_1	0.01	50	0.38	2.90	0.0029	0.0120	31.1	< 1	310.5	0
ℓ_1	0.01	100	0.39	2.72	0.0019	0.0074	46.9	< 1	619.0	0
ℓ_1	0.01	250	0.39	2.37	0.0008	0.0030	90.6	16.6	1,474.1	0
ℓ_1	0.01	500	0.40	1.89	0.0004	0.0014	116.5	76.1	2,437.1	0
ℓ_1	0.01	1,000	0.40	1.30	0.0002	0.0005	159.7	306.6	3,876.7	0
		all (avg)	0.39	2.34	0.0017	0.0073	76.8	66.7	1,481.4	0
KL	-	30	0.38	2.97	0.0028	0.0193	24.8	< 1	545.8	0
KL	-	50	0.62	3.13	0.0030	0.0125	31.9	< 1	707.6	0
KL	-	100	0.85	2.93	0.0022	0.0077	27.5	2.5	668.6	0
KL	-	250	1.64	2.85	0.0022	0.0040	13.6	1.2	524.1	1
KL	-	500	1.35	1.89	0.0008	0.0014	6.6	2.4	618.0	1
KL	-	1,000	0.90	1.19	0.0003	0.0005	5.8	16.4	1,094.2	5
		all (avg)	0.96	2.49	0.0019	0.0076	18.4	3.9	693.1	7

should be taken of (e.g., by using another NLP solver or the Frank-Wolfe algorithm as proposed by [86]).

B.1.7 Impact of reducing the size of the choice models on predictive accuracy

To assess the impact of reducing the size of the choice models, we performed further experiments where we investigate the prediction accuracy when using smaller choice models, for both the CG-GPT and the CG-LS. We trained the choice models with the two approaches and a time limit of 12 hours. Then, no matter whether the training converged or not, we have kept only the *max* K preference sequences with highest probabilities and then re-solved the restricted Master Problem to re-estimate the new probabilities.

In Table B.9 we report the dimension K of the resulting choice models for *max* $K = full$, where we keep the full (original) set of preferences sequences, *max* $K = 0.5$, where we keep only half of them, as well as *max* $K = 100$ and *max* $K = 10$. As one may expect, training and test errors increase for both approaches when decreasing the size of the corresponding choice models. However, the loss in predictive accuracy is much bigger for CG-LS. In fact, even with only 100 preference sequences, the CG-GPT is able to achieve smaller average test errors than the the CG-LS with the full set of behaviors.

Table B.9 Training and Test errors (averaged over 10 random instances) for the choice models obtained by the CG-LS and the CG-GPT, keeping only the *max K* best columns

		CG-GPT				CG-LS			
			Train	Test	# inst.		Train	Test	# inst.
	N	K	err.	err.	unsolved	K	err.	err.	unsolved
max K = full	30	170.7	0.37	2.88	0	177.5	0.39	3.74	0
	50	310.5	0.38	2.90	0	337.3	0.40	4.22	0
	100	619.0	0.39	2.72	0	667.7	0.40	4.41	0
	250	1,474.1	0.39	2.37	0	1,175.0	1.11	4.88	10
	500	2,437.1	0.40	1.89	0	413.7	6.00	9.06	10
	1,000	3,876.7	0.40	1.30	0	150.8	27.68	28.27	10
	all (avg)	993.1	0.39	2.34	0	487.0	6.00	9.10	30
max $K = 0.5$	30	85.1	0.69	3.06	0	88.4	0.78	3.91	0
	50	155.2	0.67	2.95	0	168.3	0.70	4.36	0
	100	307.9	0.64	2.76	0	333.6	0.79	4.65	0
	250	736.9	0.62	2.45	0	587.2	1.83	5.37	10
	500	1,218.3	0.62	1.93	0	206.6	9.25	11.69	10
	1,000	1,938.1	0.58	1.32	0	75.0	31.87	32.13	10
	all(avg)	740.3	0.64	2.41	0	196.8	7.54	10.35	30
max $K = 100$	30	96.3	0.54	2.99	0	99.2	0.60	3.87	0
	50	92.2	1.32	3.40	0	98.7	1.78	4.87	0
	100	90.2	2.60	4.26	0	98.0	5.18	7.47	0
	250	90.4	4.27	5.87	0	99.3	11.93	12.89	10
	500	89.5	5.30	7.74	0	100.0	15.87	17.18	10
	1,000	86.8	6.75	9.93	0	100.0	30.40	30.77	10
	all (avg)	90.9	3.46	5.70	0	99.2	10.96	12.84	30
max $K = 10$	30	9.6	8.28	9.32	0	10.0	8.75	9.57	0
	50	9.6	10.37	11.02	0	10.0	11.41	12.70	0
	100	9.0	12.33	11.97	0	10.0	16.08	17.13	0
	250	8.3	13.59	14.20	0	10.0	20.87	21.42	10
	500	8.8	12.74	13.56	0	10.0	25.95	26.43	10
	1,000	9.0	10.91	14.91	0	10.0	36.45	36.49	10
	all (avg)	9.1	11.37	12.50	0	10.0	19.92	20.62	30

B.2 Additional Results for the Assortment Optimization Algorithm

In this appendix, we provide additional computational results for the proposed assortment optimization formulation.

B.2.1 Assortment optimization: Comparison with Boosting Approach.

Even though we can add the indifference constraints (5.7) and (5.8) via branch-and-cut to the assortment optimization MIP (5.4) to directly operate on a partially-ranked choice model $\sigma = (P(\sigma), I(\sigma))$, we may attempt to complete the strictly ranked products by imposing a strict order on the products in the indifference set $I(\sigma)$. Creating several of those fully-ranked preference lists at random is called *boosting*.

We define two parameters to control the total number of fully-ranked lists and to assure that a preference list σ_k with high probability λ_k yields more fully-ranked lists than a σ_k with low probability λ_k. We define n_{min} as the minimum number of lists generated for each of the original preference lists. We also define τ as a scale parameter to control the magnitude of lists generated in proportion to the value of λ_k. For each partially-ranked preference list σ_k, we generate $n_{min} - 1 + \tau\lambda_k$ lists in which the products in the indifference set are ordered at random.

Table B.10 Assortment optimization approximation via boosting compared to AO-B&C algorithm (averaged over 100 random instances with $N = 100$ and $r = 0.5$).

			Expected revenu			GT revenu		
				Gap % from			Gap % from	
		time	avg	AO-B&C		avg	AO-B&C	
	K	(min)	rev	avg	std-dev	rev	avg	std-dev
AO-B&C	615.0	2.6	86.25	-	-	86.77	-	-
$\tau = 10$	1,845.1	5.9	86.90	1.03	0.78	86.22	0.74	1.02
$\tau = 50$	1,854.2	6.4	86.87	0.94	0.76	86.29	0.61	0.69
$\tau = 100$	1,872.9	6.7	86.69	0.80	0.66	86.27	0.66	0.74
$\tau = 500$	2,141.1	9.7	86.57	0.56	0.39	86.59	0.33	0.49
$\tau = 1,000$	2,591.3	15.1	86.39	0.39	0.32	86.61	0.29	0.48
$\tau = 5,000$	6,546.9	98.0	86.29	0.21	0.18	86.74	0.14	0.24
$\tau = 10,000$	11,541.4	318.5	86.26	0.17	0.14	86.75	0.16	0.33

Table B.10 compares the sizes K of the generated choice models and the average revenues of the optimized assortments for the exact approach AO-B&C and the boosting approach AO-Boost with different values for parameter τ (with $n_{min} = 3$). In AO-B&C, the indifference constraints are added via user callbacks by adding the first 2,500 violated constraints at each callback. Revenues are reported as average values of the expected revenue, which refers to the objective function value of the optimization problem, and as the revenue as evaluated by the ground-truth model. For both revenue types, the table reports the average deviation of the revenue given by the boosting approach from the revenue given by the exact AO-B&C approach, as well as the corresponding standard deviation.

As τ increases, both the expected and the GT revenues provided by AO-Boost get closer to the exact revenues as given by AO-B&C. However, the number of fully-ranked preference lists generated in the final choice model quickly grows (and, as a consequence, the computing times as well). The resulting optimization models are therefore too difficult to solve and not competitive with the exact branch-and-cut approach AO-B&C. This has been confirmed in further experiments, comparing the two optimization approaches AO-B&C and AO-Boost (with $\tau = 100$) for instances of different sizes N. Table B.11 reports, for each approach, the average size K of the choice model, the average computing times to solve the optimization model and the average revenue as computed by the ground-truth model. All reported values are averages over the instances solved by each method. We also report in the column "# inst unsolved" the number of instances that either ran out of memory or could not be solved within the given time limit of 12 hours. For each instance size N, we also report the optimal ground-truth revenue given the generating MMNL model [93]. The results are coherent with the findings in Table B.10. They additionally show that the B&C approach is superior both in terms of speed and assortment revenues. Further, while the B&C reports some unsolved instances due to the given time limit, the boosting approach additionally encounters problems hitting the memory limits, given the high number of generated columns on larger instances.

Table B.11 Comparison B&C vs. Boosting for Assortment optimization of choice models generated with the CG-GPT.

		CG-GPT - AO B&C				CG-GPT - AO-Boost			
N	opt GT revenue	K	time (min)	GT revenue	# inst. unsolved	K	time (min)	GT revenue	# inst. unsolved
30	78.73	170.7	<1	78.05	0	561.8	<1	77.93	0
50	84.18	310.5	<1	83.84	0	965.8	<1	82.93	0
100	88.51	619.0	3.6	88.24	0	1,884.4	6.8	87.91	0
250	91.61	1,474.1	81.5	91.39	0	4,445.0	456.3	90.84	0
500	93.51	2,374.9	515.0	93.30	2	-	-	-	10
1,000	95.48	-	-	-	10	-	-	-	10
all (avg)	88.67	989.8	117.9	86.96	12	1,964.3	115.9	84.90	20

B.2.2 Assortment optimization based on choice models trained with different loss functions

Table B.12 investigates the impact of the loss function used for estimating a partially-ranked choice model on the assortment optimization task. We compared the KL divergence against the ℓ_1 error with different training accuracy thresholds ($\epsilon_0 \in \{0.1, 0.01\}$). The results show that, besides being competitive in terms of revenue of the generated assortments, the model that minimizes the KL divergence is able to solve a number of instances similar to the model that minimizes the ℓ_1 error with $\epsilon_0 = 0.1$ (thanks to a relatively sparse choice model obtained on large instances). Note that all unsolved instances are due to the fact that the underlying choice models could not be estimated in the given time limit (often due to convergence issues of the KL based approach). Further, as mentioned in the discussion of Table B.8, additional care should be taken of when minimizing the KL loss function to guarantee convergence of the estimation algorithm on large instances.

Table B.12 Assortment optimization on choice models generated by the CG-GPT with different objective functions and training accuracy thresholds (averaged over 10 random instances, $M = 20$).

		CG-GDT - AO B&C				CG-GDT - AO B&C				CG-GDT - AO B&C			
		KL				ℓ_1, $\epsilon_0 = 0.01$				ℓ_1, $\epsilon_0 = 0.1$			
N	opt GT revenue	K	time (min)	GT revenue	# inst. unsolved	K	time (min)	GT revenue	# inst. unsolved	K	time (min)	GT revenue	# inst. unsolved
30	78.73	545.8	< 1	78.16	0	170.7	< 1	78.05	0	32.5	< 1	77.83	0
50	84.18	707.6	1.1	83.77	0	310.5	< 1	83.84	0	48.2	< 1	83.47	0
100	88.51	668.6	7.5	88.19	0	619.0	3.6	88.24	0	71.9	< 1	87.93	0
250	91.61	524.1	28.6	91.43	1	1,474.1	81.5	91.39	0	171.1	2.4	91.41	0
500	93.51	618.0	164.5	93.39	1	2,374.9	515.0	93.30	2	389.2	14.3	93.40	0
1,000	95.48	1,094.2	728.9	92.73	5	-	-	-	10	854.8	394.6	95.40	5
all (avg)	88.7	693.1	155.2	87.95	7	989.8	117.9	86.96	12	261.3	68.6	88.24	5

B.2.3 Impact of reducing the size of the choice models on assortment quality

As discussed in Section 5.3.1 and Appendix B.1.7, partially-ranked choice models may achieve better generalization with sparser models than fully-ranked choice models. Furthermore, using a smaller set of preference sequences directly facilitates the solution of the assortment optimization problem.

In Table B.13 we study the impact of the reduced number of preference sequences on the quality of the optimized assortments. This has been done by retaining only the "max K" preference sequences with highest probabilities λ. Then, the reduced choice model has been re-solved to estimate the new probabilities. For each set of experiments, we report the average size K of the choice models used as an input, the average computing times of the assortment optimization, the average of the corresponding ground-truth revenues and the average optimality gaps reached after the time limit of 12 hours. The results suggest that it may be advantageous to limit the number of preference sequences generated by the CG-GPT to a reasonable number, given that predictive accuracy remains strong, and the final choice models are easier to optimize on. In particular, the best results are obtained with

Table B.13 Assortment optimization results (averaged over 10 random instances), using the CG-LS and the CG-GPT, keeping only the *max K* best columns .

		CG-GDT - AO B&C				CG-LS-AO-Compl			
			time		opt		time		opt
	N	K	(min)	GT rev	gap %	K	(min)	GT rev	gap %
max K = full	30	170.7	3.7	78.05	0.00	177.5	<1	76.37	0.00
	50	310.5	18.8	83.84	0.00	337.3	<1	81.76	0.00
	100	619.0	216.0	88.24	0.00	667.7	<1	87.31	0.00
	250	1,474.1	4890.2	91.39	0.00	1175	16.4	90.65	0.00
	500	2,437.1	557.5	93.38	0.30	413.7	13.7	90.92	0.00
	1,000	3,876.7	720.2	93.00	5.04	150.8	10.2	92.75	0.00
	all (avg)	993.1	1067.7	86.94	0.90	394.2	<1	86.63	0.00
max K = 0.5	30	85.1	<1	78.18	0.00	87.4	<1	76.90	0.00
	50	155.2	<1	83.87	0.00	166.7	<1	81.71	0.00
	100	307.9	<1	88.25	0.00	331.1	<1	87.02	0.00
	250	736.9	23.9	91.39	0.00	583.3	5.3	90.54	0.00
	500	1,218.3	238.9	93.40	0.00	206.6	6.8	90.13	0.00
	1,000	1,928.6	745.0	93.33	3.70	75	4.6	90.45	0.00
	all(avg)	500.7	168.1	87.02	0.62	196.8	2.9	86.12	0.00
max K = 100	30	96.3	<1	78.05	0.00	99.2	<1	77.09	0.00
	50	92.2	<1	83.85	0.00	98.7	<1	80.76	0.00
	100	90.2	<1	88.30	0.00	98	<1	86.14	0.00
	250	90.4	<1	91.39	0.00	99.3	<1	88.46	0.00
	500	89.5	1.4	93.40	0.00	100	2.2	87.84	0.00
	1,000	86.8	4.2	95.42	0.00	100	5.5	91.56	0.00
	all (avg)	90.9	1.1	88.40	0.00	99.2	1.4	85.31	0.00
max K = 10	30	9.6	<1	76.80	0.00	10	<1	73.47	0.00
	50	9.6	<1	83.42	0.00	10	<1	72.81	0.00
	100	9.0	<1	88.18	0.00	10	<1	78.86	0.00
	250	8.3	<1	91.39	0.00	10	<1	76.02	0.00
	500	8.8	<1	93.40	0.00	10	<1	79.04	0.00
	1,000	9.0	<1	95.42	0.00	10	<1	84.48	0.00
	all (avg)	9.1	<1	88.10	0.00	10	<1	77.45	0.00

max $K = 100$. In contrast, any attempts to reduce the number of preference sequences for the CG-LS based choice models results in a loss of predictive accuracy, which then translates into worse assortments revenues.

B.2.4 Hardness of assortment optimization

Even though the principal contribution of this paper lies in the proposal of the partially-ranked choice model along with an efficient estimation procedure, the final assortment optimization relies on an adaptation of MIP (5.4). Throughout all of our experiments, the generated choice models have been well handled by our assortment optimization algorithm. In theory, however, the difficulty of solving MIP (5.4) strongly depends on the structure of the choice model used. In this regard, [88] use a reduction from the Maximum Independent Set problem to prove the difficulty of assortment optimization over preference lists. Using such reduction, the authors are able to identify choice model structures that make the solution particularly difficult. Even though our proposed estimation procedure has not, and is unlikely to produce choice models of such structure, we here numerically explore how such choice model structures could impact the scalability of the resulting optimization problem.

In particular, following the work of [88], we investigate the impact of different structural assumptions about the preference sequences. Given N products, we generate N customer types using one of the following strategies:

- *constant-degree followed by 0*: each customer type consists of 4 ranked items followed by the no-purchase option 0. These instances correspond to a constant-degree graph and should be easy to optimize on. It is worth noting that this is different from the way we generate the preference sequences in our estimation method, given that in our case non-favorite items in the ground-truth model can still be sold with non-negative probability.

- *constant-degree followed by random permutation*: each customer type has 4 favourite items. In the corresponding preference sequence, such items are followed by a random permutation of the remaining options. This way of generating preference sequences is more similar to those generated by the CG-GPT, since a customer is allowed to buy a non-favorite product with non-negative probability.

- *random permutations*: we do not make any assumptions or integrate any prior knowledge about/into the structure of the preference sequences.

- *neighborhood-based*: based on the DIMACS Maximum Independent Set instances [3] the 0-option is added to the list of vertices; we then generate preference sequences that are based on the concept of product neighborhoods in the corresponding graph (see [88]).

Table B.14 AO-Compl computation times to solve MIP (5.4) to optimality, when choice models have different structures. (Time limit of 12 hours)

Preference structure	$N = K$	computing time (min)
constant degree	451	2.7
	596	6.5
	761	11.7
	946	24.6
constant degree + random permutation	451	21.1
	596	35.2
	761	116.0
	946	361.5
random permutation	451	20.8
	596	40.2
	761	175.0
	946	433.9
neighborhood-based	451	34.5
	596	298.2
	761	653.1
	946	720

For each of the four cases, a revenue i was assigned to each of the products $i \in \{1, ..., N\}$, while the customer arrival probabilities have been randomly drawn from the unit simplex. A total of 10 random instances has been generated for different instance sizes $N = K$. The optimization for each instance has been limited to a total of 12 hours computing time. Table B.14 reports the average computing times required to solve the optimization models based on the corresponding type of instances. As one may expect, constant-degree instances are the easiest to solve. In line with the conclusions of [88], neighborhood-based instances seem to be the most difficult to solve, hitting the given time limit on large instances. Finally, we note that the computing times that seem to be the closest to those observed in the experiments for our method (see Table 5.3) are those that are randomly generated. This suggests that the choice models generated by our estimation procedure do not hold the same structural properties of the neighborhood-based choice models, which are more difficult to optimize on.

[3]Available at http://sites.nlsde.buaa.edu.cn/$\tilde{\ }$kexu/benchmarks/graph-benchmarks.htm

The instances used in the experiments above can be found online at:
`https://cerc-datascience.polymtl.ca`

APPENDIX C ON THE ESTIMATION OF DISCRETE CHOICE MODELS TO CAPTURE IRRATIONAL CUSTOMER BEHAVIORS

C.1 “The Economist” Choice Experiment from [102]

Example C.1.1 *In this experiment, students where asked to choose among different subscription plans for the magazine “The Economist”. In particular, the following three options were used : (1) Only version only, priced 50\$, (1) Printed version only, with a price of 125\$, and (3) Printed and Online subscription, at a price of 125\$. Table C.1 reports the market share of the various options in choice scenarios S_1, where only options {1,3} were offered to students, and S_2, where students where able to choose among the three options {1,2,3}. This experiment exhibits a violation of the regularity assumption, since the probability of choosing option (3) increases from 32% to 84% when option (2) is added to the offer set. Hence, no model belonging to the RUM class can perfectly fit this dataset.*

Table C.1 Predicted shares of three camera models in choice scenarios S_1, where respondents must choose between alternatives $\{1, 2\}$, and S_2, where option (3) is added to the offer set

		Market Share	
Versions	Price (\$)	S_1	S_2
(1) Online	59	.68	.16
(2) Printed	125	–	.0
(3) Printed & Online	125	.32	.84

The choice phenomena reported in Table C.1 is an example of the so-called *decoy effect*. In fact, option (2) is clearly “dominated” by option (3) in terms of attractiveness, which allows, for the same price, to obtain both the Printed *and* Online versions of the magazine. Options perceived inferior in terms of quality and/or price, i.e., decoy options, are often used in Marketing to increase the perceived attractiveness of other products.

In Table C.2 we report a GSP choice model perfectly fitting the choice outcome of the experiment in Example C.1.1. Specifically, we report the only three behaviors $C_k(\sigma_k, i_k)$, with $k = 1, 2, 3$, with non-zero probability, and how choice are determined in the two choice scenarios. Finally, Table C.3 reports the choice probabilities predicted by the model, which match the observed ones from Table C.1.

Table C.2 GSP model from [2] explaining the choice outcomes of Example C.1.1. For each $\sigma_{k,S}$, we highlight in bold the chosen item $j : \sigma_{k,S}(j) = i_k$.

Customer Type			Probability	$S_1 = \{1, 3\}$	$S_2 = \{1, 2, 3\}$
k	σ_k	i_k	λ_k	σ_{k,S_1}	σ_{k,S_2}
1	$(3, 1, 2)$	1	0.16	$(\mathbf{3}, 1)$	$(\mathbf{3}, 1, 2)$
2	$(2, 1, 3)$	2	0.16	$(1, \mathbf{3})$	$(2, \mathbf{1}, 3)$
3	$(2, 3, 1)$	2	0.68	$(3, \mathbf{1})$	$(2, \mathbf{3}, 1)$

Table C.3 Predicted shares of three camera models in choice scenarios S_1, where respondents must choose between alternatives $\{1, 2\}$, and S_2, where option (3) is added to the offer set

		Predicted Share	
Versions	Price ($)	S_1	S_2
(1) Online	59	$\lambda_3 = .68$	$\lambda_2 = .16$
(2) Printed	125	–	0
(3) Printed & Online	125	$\lambda_1 + \lambda_2 = .32$	$\lambda_1 + \lambda_3 = .84$

C.2 Regularity violation of the no-purchase option

The Generalized Stochastic Preference choice model as defined by [2] does not account for violations of the regularity assumption for the no-purchase option [see 2, Lemma 1]. Given two offer sets $S \subseteq S' \subseteq \mathcal{N}$, in particular, the authors show that every customer choosing the no-purchase option from S', by definition, must choose the no-purchase option from S as well. However, we can circumvent such limitation by allowing a customer type $C_k(\sigma_k, i_k)$ to rank the no-purchase option in σ. Consider, for example, two offer sets $S = \{0, 1, 2\}$ and $S' = \{0, 1, 2, 3\}$, where $S \subset S'$, and a customer type $C_1\big((3\ 0\ 1\ 2), 1\big)$. Her choice behavior is reported in Table C.4. It is easy to see that, by introducing the option 3 in the offer set, we can increase the probability of option 0 being chosen and, thus, of the customer leaving without any purchase.

Table C.4 Choice behavior of customer $C\big((3\ 0\ 1\ 2), 1\big)$ faced with two different offer sets.

S	$\sigma_{1,S}$	Choice
$\{0, 1, 2\}$	$(0\ 1\ 2)$	1
$\{0, 1, 2, 3\}$	$(3\ 0\ 1\ 2)$	0

C.3 Details on the implementation of PCMC

In this section, we elaborate on the implementation details of the PCMC choice model. The model is trained by Maximum Likelihood Estimation, and a Sequential Least SQuares Programming (SLSQP) solver [133] is used to optimize the corresponding objective function, which is concave in general. The authors suggest to use additive-smoothing to avoid some numerical issues involved in the training of the model. In particular, given an offer set of size $|S|$ and an additive smoothing parameter α, the probability of choosing alternative j is computed at training time as

$$P(j|S) = \frac{T_{jS} + \alpha}{T_S + \alpha|S|},$$

where T_S is the number of training samples showing offer set S, and T_{jS} the number of times alternative j is chosen from the offer set S. In Table C.5, we investigate the change in performance due to different stopping criteria and values of parameter α. In particular, we implemented stopping criteria based on

1. The maximum number of iterations to be performed by the solver: this is set to 25, which is the default value in the code provided by the authors. The corresponding results are reported in column "PCMC-25"
2. The absolute change in the objective function between two consecutive iterations: the algorithm is stopped when this change is smaller than 10^{-6}, and the corresponding results are reported in column "PCMC-∞".

Moreover, for each stopping criterion, we compared the performance obtained by using different amounts of additive-smoothing in the training set. Specifically, column "Crossval" reports the average generalization error obtained using 5-folds crossvalidation to select the best $\alpha \in \{0, 0.01, 0.1, 1, 5, 10\}$. Column "None" corresponds to $\alpha = 0$, for which no additive smoothing was used.
The average L_1 test error has been reported over instances grouped by ground-truth models and number of customers types, indicated in parenthesis in the first column. The value of column "% irrat" further divides each group based on the characteristics of the ground-truth model generating the corresponding set of instances. For Halo-MNL instances, this column indicates the amount of pairwise interactions among products, while, for GSP instances, it indicates the percentage of irrational behaviors. We observe that limiting the number of iterations seems to be having a major impact on the predictive accuracy of the resulting choice model. Our intuition is that allowing the solver to proceed until convergence is reached may

Table C.5 Average L_1 test errors for different PCMC implementations under various ground truth models. Each line averages over instances generated with different number of training offer sets (10,20 and 50) and transactions (3,000 and 50,000).

Irrational instances		PCMC -25		PCMC - ∞	
	% irrat	Crossval	None	Crossval	None
Halo-MNL(1)	10	**0.2595**	0.2610	0.2897	0.2941
	25	**0.3756**	0.3804	0.3933	0.3937
	avg (all)	**0.3175**	0.3207	0.3415	0.3439
Halo-MNL(10)	10	**0.1997**	0.2195	0.2348	0.2545
	25	**0.2430**	0.2518	0.2873	0.3010
	avg (all)	**0.2214**	0.2356	0.2610	0.2778
GSP(10)	10	0.4293	**0.4273**	0.4615	0.4625
	20	**0.4622**	0.4625	0.4927	0.4918
	50	**0.5283**	0.5312	0.5668	0.5711
	avg (all)	**0.4733**	0.4737	0.5070	0.5084
GSP(100)	10	**0.2790**	0.2890	0.3223	0.3386
	20	**0.2880**	0.3009	0.3354	0.3478
	50	**0.3283**	0.3394	0.3868	0.3976
	avg (all)	**0.2984**	0.3098	0.3481	0.3613
avg (all)		**0.3500**	0.3568	0.3887	0.3967
Rational instances					
MNL	-	**0.1351**	0.1404	0.1930	0.2012
MMNL	-	**0.1381**	0.1529	0.1970	0.2136
RB(10)	-	**0.3840**	0.3841	0.4028	0.4065
RB(100)	-	**0.2741**	0.2793	0.3167	0.3292
avg (all)		**0.2328**	0.2392	0.2774	0.2876

end up in overfitting the training set. We also notice that the gain in performance obtained by using additive-smoothing is not significant in general. To confirm whether the deterioration in performance of PCMC-∞ is actually due to overfitting, in Table C.6 we further compare the two variants on a set of instances where $50,000$ training samples have been generated, and we investigate the impact of the number of training offer sets on the resulting choice model. Confirming our previous hypothesis, we notice that when significant amount of training data is available, both in terms of number of training samples and number of training offer sets M, the risk of overfitting decreases and a better fit at training time translates in a significant improvement in generalization error. Nevertheless, also for this set of instances, i.e., with $M = 50$ offer sets and $50,000$ samples are available at training time, the performance of the best PCMC variant is worse than the one of the GPT-based approaches. At this point, one

Table C.6 Average L_1 test errors for different PCMC implementations under various ground truth models, on instances with 50,000 choice samples available for training. Instances are further divided based on the number of offer sets M observed during training.

Irrational		PCMC - 25		PCMC - ∞	
	M	Crossval	None	Crossval	None
Halo-MNL	10	**0.3091**	0.3097	0.3341	0.3332
	20	0.2824	**0.2801**	0.2827	0.2826
	50	0.2340	0.2329	**0.1546**	0.1570
GSP	10	**0.4859**	0.4873	0.5347	0.5340
	20	**0.4052**	0.4069	0.4300	0.4322
	50	0.3355	0.3383	**0.2817**	0.2827
avg (all)		0.3677	0.3688	**0.3667**	0.3675
Rational					
(M)MNL	10	**0.1212**	0.1216	0.1518	0.1497
	20	**0.0921**	0.0924	0.1231	0.1294
	50	0.0733	0.0704	**0.0659**	0.0662
RB	10	**0.4187**	0.4291	0.4897	0.4997
	20	0.3530	**0.3429**	0.3794	0.3893
	50	0.3072	0.2961	0.2099	**0.2086**
all (avg)		**0.3490**	0.3497	0.3494	0.3506

may wonder whether more adaptive stopping criteria may be used instead of fixing ahead the maximum number of iterations. However, further experiments revealed that fixing the number of iterations to 25 worked better on average than other stopping criteria based on

- The relative change in the objective function between consecutive iterations ($< 1\%$),
- The maximum absolute change in the predicted probabilities over all training offer sets (< 0.001),
- The maximum absolute change in the value of the parameters of the PCMC choice model,
- The maximum number of iterations set to 100.

We thus avoid reporting the set of results corresponding to such stopping criteria, and use the PCMC-25 variant with no additive smoothing in the rest of our experiments. In particular, this is also the PCMC implementation used for the experiments reported in Section 6.5.1 and Section 6.5.2.

C.4 Additional Numerical Results

C.4.1 Learning statistics

Table C.7 Statistics describing choice models learned by GPT-based approaches.

Instances		Max ranked			Max irrat level			Prob. Irrat. Customer		
	%irrat	GPT-R	GPT-I	GPT-IC	GPT-R	GPT-I	GPT-IC	GPT-R	GPT-I	GPT-IC
MNL	0	2.8	2.9	2.1	0	1.7	1.1	0	0.45	0.39
Halo-MNL(1)	10	4.3	4.4	2.7	0	2.9	1.7	0	0.51	0.40
Halo-MNL(1)	25	4.8	5.8	3.3	0	3.7	2.1	0	0.61	0.49
MMNL	0	2.3	2.4	2.0	0	1.3	1.0	0	0.43	0.40
Halo-MNL(10)	10	2.9	2.8	2.0	0	1.7	1.0	0	0.51	0.43
Halo-MNL(10)	25	3.5	3.4	2.4	0	2.2	1.4	0	0.56	0.47
RB(10)	0	4.8	5.5	3.7	0	3.7	2.2	0	0.34	0.30
GSP(10)	10	4.9	5.6	3.7	0	3.8	2.3	0	0.40	0.35
GSP(10)	20	4.9	6.1	3.9	0	4.2	2.5	0	0.45	0.38
GSP(10)	50	5.0	6.9	4.1	0	4.9	2.8	0	0.56	0.45
RB(100)	0	3.3	3.4	2.4	0	2.1	1.4	0	0.49	0.40
GSP(100)	10	3.3	3.5	2.4	0	2.3	1.4	0	0.52	0.44
GSP(100)	20	3.5	3.7	2.5	0	2.5	1.4	0	0.56	0.47
GSP(100)	50	3.8	4.4	2.6	0	3.1	1.5	0	0.62	0.50
(All) Mean	-	4.0	4.6	2.9	0	3.1	1.8	0	0.51	0.43
(All) Median	-	4	4	3	0	3	2	0	0.51	0.43
(All) Max	-	9	10	9	0	9	8	0	0.96	0.96

We now elaborate on the impact of the irrationality level of an instance on the choice models learned by GPT-based approaches. In particular, Table C.7 groups instances based on ground-truth models and number of customer types used to generate the data. For each group, we sort Halo-MNL and GSP instances by increasing levels of irrational interactions and behaviors, respectively, in the ground-truth model. We then report, for each approach, its maximum number of strictly ranked products, the maximum irrationality level of its customer types, and the total probability of a customer being irrational. This, in particular, corresponds to the sum of the probabilities of irrational customer types in the learned choice model.

Confirming the results from [73], all approaches strictly rank only a relatively small number of products on average, but are able to capture high order interactions when needed, by strictly ranking up to 10 products in specific cases. Also, for the irrational approaches, i.e., GPT-I and GPT-IC, both the maximum level of irrationality and the probability of a customer being irrational seem to generally increase with the level of irrationality of the ground-truth model. However, we note that on rational instances, GPT-I and GPT-IC still predict a customer being irrational with a relatively high probability. While, as observed in Section 6.5.1.2, high number of irrational customer classes may still result in close-to-rational behavior at the population level, this may justify the superior performance of GPT-R on such instances.

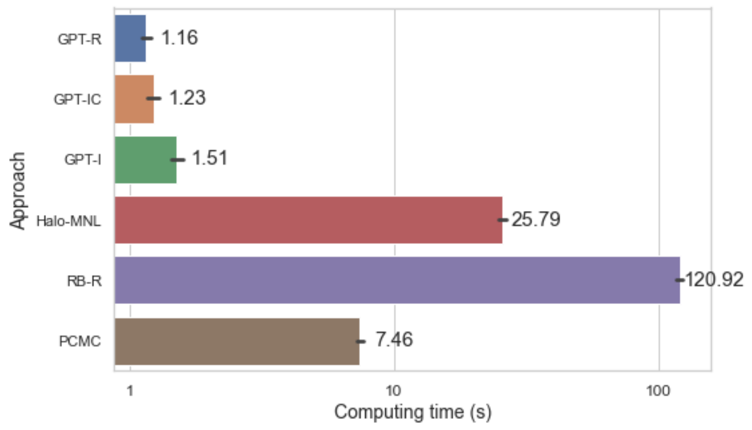


Figure C.1 Average Computing times (seconds) for the various approaches over all synthetic instances.

We conclude this set of analysis by diving into the computational aspect of the GPT-estimation procedure. In particular, as argued in Section 6.4.3, the computational cost one has to pay for extending the algorithm from [73] to include irrational behaviors, stems from the fact that splitting each node (i.e., behavior) of the search-tree results in $|I(\sigma)| \cdot |P(\sigma)| \in O(N^2)$ new sub-behaviors, compared to the $|I(\sigma)| \in O(N)$ ones of the rational case. However, as observed from Table C.7, the GPT procedure tends to discover customer types whose number of strictly ranked products $|P(\sigma)|$ is rather small. In practice, this makes the discovery of irrational behaviors computationally efficient. In this regard, we report in Figure C.1 the computing times of various approaches averaged over all synthetic instances, confirming the computational effectiveness of GPT-based approaches.

C.4.2 Impact of the amount of available data on the Loss of Rationality

In this section we show that the Loss of rationality (LoR) can be influenced by factors other than the presence of irrational consumer behaviors. In particular, Figure C.2 shows that lower number of transactions available for training tend to lead to higher loss of rationality. Indeed, many interactions that may appear as irrational for small number of transactions, may be due to sampling noise and thus tend to disappear when more transactions become available.

Another source of increase in the LoR of an instance stems from the number of unique

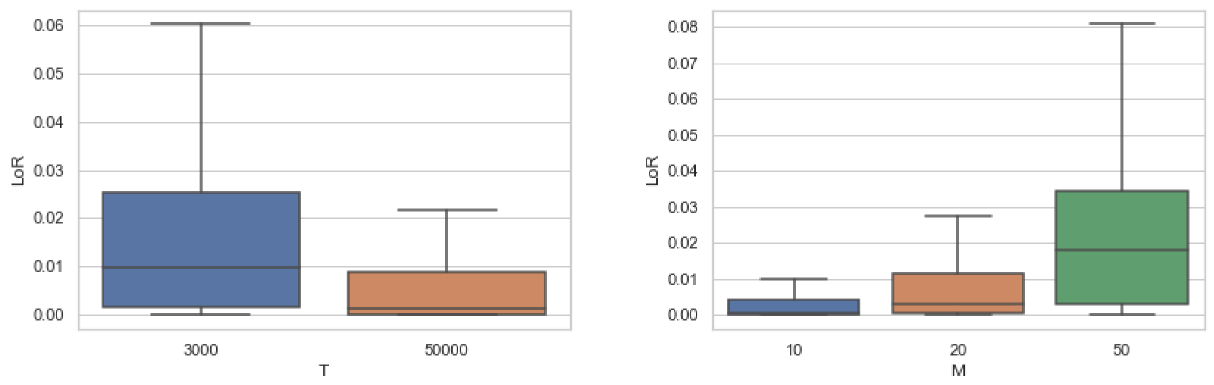


Figure C.2 Impact of the number of transactions T (Left) and unique assortments M (Right) available for training on the Loss of Rationality of instances.

assortments available for training. Indeed, consider the extreme case in which transactions are available for only one assortment. Such data, clearly, can be perfectly fit by any choice (or independent demand) model, since no substitution nor halo effects can be observed in this case. Hence, its LoR will be zero. As the number of assortments increases, more patterns of interactions among products become available, thus resulting in possibly complex choice probability distributions that may contain irrational interactions as well. Datasets with many training assortments are thus more likely to be associated with high LoR.

The two observations above shed light on the limitation of the LoR metric as a tool for model selection, since no *a priori* consideration can be made about an "acceptable" level of LoR beyond which one may consider to go beyond RUM.

C.4.3 Impact of the irrationality level of GSP customer types.

As observed in Section 6.4.3, generalized stochastic preferences with different irrationality levels imply different types of interaction among products. We are thus interested in understanding how well each of the approach we implemented generalizes under different levels of customer irrationality i_{max}. To this end, in Table C.8 we group GSP instances by the number of customer types (in parenthesis) and maximum irrationality level i_{max} in the ground-truth model. While particularly high levels of irrationality may not be very common in practice, they allow us to analyze possible limitations of approaches, given the limited irrationality assumption intrinsic to the GPT-procedure.

We start by noticing that, on average, both Halo-MNL and PCMC are outperformed by rank-based approaches on this set of experiments. Also, as observed in Section 6.5.1.3, GPT-IC

Table C.8 Test errors comparison on GSP instances grouped based on the irrationality level of customer types in the ground-truth model. The metric reported is the average L1 error per offer set

Instances	i_{max}	LoR	RB-R	GPT-R	GPT-I	GPT-IC	PCMC	Halo-MNL
GSP(10)	1	0.0332	0.3131	0.2963	**0.2601**	0.2819	0.4824	0.6923
GSP(10)	5	0.0438	0.3414	0.3261	**0.3145**	0.3297	0.5208	0.7659
GSP(10)	9	0.0260	0.2436	**0.2261**	0.2351	0.2458	0.4177	0.6645
	All (Mean)	0.0343	0.2994	0.2828	**0.2699**	0.2858	0.4736	0.7075
GSP(100)	1	0.0043	0.1903	0.1654	0.1630	**0.1606**	0.3052	0.2748
GSP(100)	5	0.0057	0.1939	**0.1773**	0.1808	0.1806	0.3307	0.3006
GSP(100)	9	0.0060	0.1798	**0.1598**	0.1665	0.1642	0.2933	0.2840
	All (Mean)	0.0053	0.1880	**0.1675**	0.1701	0.1685	0.3098	0.2865
All (Mean)		0.0198	0.2437	0.2252	**0.2200**	0.2272	0.3917	0.4970
All (Median)		0.0057	0.2142	0.1898	**0.1862**	0.1918	0.3634	0.3796
All (Max)		0.2343	0.9454	0.7877	0.8087	**0.7835**	1.0035	1.5872

tends to dominate GPT-I on instances with several customer behaviors, while the opposite is true for instances with less number of customer behaviors. On such instances, GPT-IC still outperforms GPT-R for i_{max} of 1, confirming the fact that GPT-IC is more suited to capture low-order interactions.

It is also interesting to note a decrease in the Loss of Rationality of instances generated for $i_{max} = 9$ and, coherently, an improvement in the predictive accuracy of rational rank-based methods on the same set of instances. Indeed, given an offer set S and a generalized stochastic preference $C_k(\sigma_k, 9)$, it is often the case that $|\sigma_{k,S}| < i$. We recall from Section 6.3 that, in such cases, the considered customer type leaves with no purchase. Intuitively, such a behavior can be more easily approximated by a rational choice model imposing a high probability mass on the no-purchase option. This also translates into predictions that are more accurate on average than those obtained for smaller levels of irrationality i_{max}.

C.4.4 Impact of the type of positive interaction

We now investigate the difference in performance of the various approaches on Halo-MNL instances with symmetric versus asymmetric product interactions. The former scenario aims to represent the case of complementarity effects, where two products increase each other attractiveness when present in the offer set. This is the case, for example, of pasta and tomato sauce, or pancake mix and maple syrup. The latter aims to represent the decoy effect (see Appendix C.1), where an item is included in the offer set for the purpose of making another item, perceived as much better, more attractive to the customer.

In Table C.9 we reports the test errors of the each approach on classes of instances grouped by ground-truth model, number of customer types (in parenthesis), and the "Type" of interaction between items, i.e., Symmetric or Asymmetric ones. On average, GPT-IC is the best performing approach among the implemented ones. In fact, as observed in Section 6.5.1.3, the Halo-MNL choice model tends to need significant amount of data to well approximate the data-generating model. We note that, even in the case of Halo-MNL instances with one customer type, Halo-MNL seems to struggle for Asymmetric interactions in particular. This is probably due to the fact that, given its N^2 parameters, Halo-MNL tends to recover less sparse matrices of pairwise interactions, especially when only limited amount of data is available.

Table C.9 Test errors comparison on GSP instances grouped based on the irrationality level of customer types in the ground-truth model. The metric reported is the average L1 error per offer set

Instances	Type	LoR	RB-R	GPT-R	GPT-I	GPT-IC	PCMC	Halo-MNL
Halo-MNL(1)	Asymm.	0.0117	0.2293	0.2378	0.1851	**0.1806**	0.2533	0.2060
Halo-MNL(1)	Symm.	0.0229	0.3112	0.3207	0.2685	0.2643	0.3878	**0.2337**
	All (Mean)	0.0173	0.2703	0.2792	0.2268	0.2224	0.3205	**0.2199**
Halo-MNL(10)	Asymm.	0.0037	0.1508	0.1255	0.1125	**0.1095**	0.2145	0.1927
Halo-MNL(10)	Symm.	0.0042	0.1643	0.1447	0.1291	**0.1263**	0.2568	0.2028
	All (Mean)	0.0040	0.1575	0.1351	0.1208	**0.1179**	0.2357	0.1977
(All) Mean		0.0106	0.2139	0.2072	0.1738	**0.1702**	0.2781	0.2088
(All) Median		0.0031	0.1954	0.1839	0.1495	**0.1432**	0.2576	0.1943
(All) Max		0.1030	0.4963	0.5055	0.4790	**0.4436**	0.7093	0.9779

C.5 Proof of Observation 1

Observation 1 states the number of spurious positive interactions of an irrational partially-ranked preference sequence $C(P(\sigma), I(\sigma), i)$. can be as high as $\sum_{j=i-1}^{|P(\sigma)|-1} \binom{j}{i-1}$.

For illustrative purpose, note here that the strictly ranked preference list of $C(P(\sigma), I(\sigma), i)$ is a sequence $(\sigma^{-1}(1), \sigma^{-1}(2), \ldots, \sigma^{-1}(|P(\sigma)|))$, where $\sigma^{-1}(r)$ denotes the item ranked at position r in $P(\sigma)$.

Depending on the offer set, any item in $P(\sigma)$ (except for the $i-1$ first items, which will never be chosen due to irrationality level i) may be positively influenced by items ranked higher in $P(\sigma)$. Summing over all items in $P(\sigma)$ that may be subject to spurious positive interactions, we have a total of $\sum_{r=i}^{|P(\sigma)|} SI_r$ possible spurious positive interactions in $C(P(\sigma), I(\sigma), i)$, where SI_r is the number of possible spurious interactions for item $\sigma^{-1}(r)$, ranked at position r.

Further, an item $\sigma^{-1}(r)$, ranked at position r can be positively influenced by degree i in a given offer set S only when exactly $i-1$ of all items that have ranks smaller than r (i.e., $(\sigma^{-1}(1), \sigma^{-1}(2), \ldots, \sigma^{-1}(r-1))$) are present in offer set S. Among those $r-1$ items, there are exactly $\binom{r-1}{i-1}$ combinations how to select $i-1$ items. We therefore can compute $SI_r = \binom{r-1}{i-1}$. The result follows as the total number of possible spurious positive interactions is $\sum_{r=i}^{|P(\sigma)|} \binom{r-1}{i-1}$.

$\square$

APPENDIX D FURTHER DISCUSSIONS

D.1 Example of fully-ranked representation for a partially-ranked behavior.

Expanding on our discussion in Appendix B.1.1, we provide in Table D.1 an example reporting the twelve fully-ranked preferences needed to represent the partially-ranked preference $\sigma_c = \Big((1,2,3), \{4,5,6,7\}\Big)$, with four items in the indifference set.

Table D.1 Fully-ranked representation of a partially-ranked preference with four items in the indifference set.

σ_c		Fully-ranked representation											
	1	1	1	1	1	1	1	1	1	1	1	1	1
$P(\sigma_c)$	2	2	2	2	2	2	2	2	2	2	2	2	2
	3	3	3	3	3	3	3	3	3	3	3	3	3
		4	4	4	5	5	5	6	6	6	7	7	7
$I(\sigma_c)$	$\{4,5,6,7\}$	5	6	7	4	6	7	4	5	7	4	5	6
		6	5	5	7	4	4	7	7	4	6	6	5
		7	7	6	6	7	6	5	4	5	5	4	4
λ	1	0.083	0.083	0.083	0.083	0.083	0.083	0.083	0.083	0.083	0.083	0.083	0.083