Universidad de Buenos Aires
Facultad de Ciencias Exactas y Naturales
Departamento de Computación

A Comparative Empirical Study of Discrete Choice Models

Tesis de Licenciatura en Ciencias de la Computación

Agustín Garassino

Director: Gustavo Vulcano

Codirector: Gerardo Berbeglia

Buenos Aires, 2020

UN ESTUDIO COMPARATIVO Y EMPÍRICO SOBRE MODELOS DE ELECCIÓN DISCRETA

La estimación de demanda es una tarea fundamental en gestión de operaciones y revenue management, brindando los datos necesarios para control de inventario, diseño de assortment y modelos de pricing. La tarea es particularmente difícil en contextos operativos donde la disponibilidad de los productos varía a lo largo del tiempo, induciendo la sustitución por parte de los consumidores. Además del clásico modelo multinomial logit (MNL) y sus extensiones (ej: nested logit, mixed MNL), nuevos modelos de demanda fueron propuestos (ej: modelo de Markov Chain) y otros fueron recientemente revisitados (ej: el ranked-list y el exponomial). Al mismo tiempo, nuevos algoritmos fueron desarrollados para facilitar su estimación (ej: generación de columnas, expectation maximization).

En este trabajo se realizó un estudio sistemático y empírico de múltiples modelos de demanda basados en elección discreta, y de distintos algoritmos de estimación, incluyendo máxima verosimilitud y cuadrados mínimos. Con un exhaustivo conjunto de experimentos numéricos sobre datos sintéticos, semi-sintéticos y reales, se proveen estadísticas comparativas del poder predictivo y la performance económica de una amplia colección de modelos. A su vez se caracterizaron los entornos operativos adecuados para su implementación. Adicionalmente, se realizó un resumen de todos los modelos evaluados.

Palabras claves: estimación de demanda, preferencias de los consumidores, comportamiento de consumidores, estimación de máxima verosimilitud, estimación de cuadrados mínimos

A COMPARATIVE EMPIRICAL STUDY OF DISCRETE CHOICE MODELS

Choice-based demand estimation is a fundamental task in retail operations and revenue management, providing necessary input data for inventory control, assortment and price optimization models. The task is particularly difficult in operational contexts where product availability varies over time and customers may substitute into the available options. In addition to the classical multinomial logit (MNL) model and extensions (e.g., nested logit, mixed MNL), new demand models have been proposed (e.g., the Markov chain model) and others have been recently revisited (e.g., the rank list-based and exponomial models). At the same time, new computational approaches were developed to ease the estimation function (e.g., column generation, EM algorithms).

In this work, a systematic, empirical study of different choice-based demand models and estimation algorithms is conducted, including both maximum likelihood and least squares criteria. Through an exhaustive set of numerical experiments on synthetic, semi-synthetic and real data, comparative statistics of the predictive power and derived revenue performance of an ample collection of choice models is provided. Furthermore, operational environments suitable for different model/estimation implementations are characterized. A survey of all the discrete choice models evaluated is also provided.

Keywords: demand estimation, consumer preferences, choice behavior, maximum likelihood estimation, least squares estimation.

ACKNOWLEDGEMENTS

This work was partially funded with a grant from the Centre for Business Analytics at Melbourne Business School. This support is gratefully acknowledged. Special thanks are given to the University of Melbourne's High Performance Computer and Cloud Hybrid Spartan and to its friendly staff. Thanks are also due to Andrea Rotnitzky (UTDT), Srikanth Jagabathula (NYU), Ashwin Venkataraman (UT Dallas), Fatma Kılınç-Karzan (CMU), and Francisco Cisternas (CUHK) for very helpful and valuable discussions. Finally, the suggestion to incorporate the Markov chain model with rank-2 matrix from Guillermo Gallego is greatly appreciated.

CONTENT

1. Introduction 1
   1.1 Contributions 3

2. Discrete choice basics 6
   2.1 Random utility models 6
   2.2 A survey of the discrete choice models benchmarked 7
      2.2.1 Independent demand 8
      2.2.2 Multinomial logit 8
      2.2.3 Exponomial 9
      2.2.4 Latent class MNL 10
      2.2.5 Nested logit 11
      2.2.6 Mixed logit 13
      2.2.7 Markov chain 14
      2.2.8 Rank list-based 16
   2.3 Summary of relevant model features 19
   2.4 Estimation 20
   2.5 Performance metrics 22

3. Package implementation 24
   3.1 Chosen language 24
   3.2 Code architecture 24
      3.2.1 Transactions 25
      3.2.2 Models 25
      3.2.3 Estimation methods 27
      3.2.4 Optimization problems 27
      3.2.5 Classes diagram and relationships 28
   3.3 Online availability 29

4. High-performance computing 30
   4.1 Computational load 30
   4.2 High-performance computing 30
   4.3 Spartan HPC 30
   4.4 Instance encoding 32

4.5 Python utility . . . 33
4.6 Reduced computational time . . . 33

5. Performance under synthetic preferences . . . 35
5.1 Description of instances . . . 35
5.1.1 Random instances . . . 35
5.1.2 Structured instances . . . 37
5.2 Results . . . 38
5.2.1 Impact of volume of historical data . . . 39
5.2.2 Impact of structured preferences . . . 40
5.2.3 Impact of the consistency of the underlying preferences . . . 41
5.2.4 Impact of the assortment size . . . 43
5.2.5 Computational time of the estimation process . . . 44

6. Performance under real preferences . . . 47
6.1 Hotel data . . . 47
6.2 Sushi dataset . . . 48
6.2.1 Estimation results . . . 49

7. Performance under boundedly rational preferences . . . 53

8. Conclusions . . . 56

Appendix 1. Implementation details . . . A1
A1 Least squares estimation . . . A1
A2 Coding of the methods . . . A2

Appendix 2. Maximum likelihood under Markov chain: Direct optimization vs EM A3

Appendix 3. Supplement to synthetic data experiments . . . A4
A1 Maximum likelihood . . . A6
A2 Least squares . . . A10
A3 Consistency of preferences: a comparison between 2 lists and 100 lists . . . A14
A4 Latent class ground truth . . . A14
A5 A ground-truth with prices . . . A17

1. INTRODUCTION

The operations management (OM) field has traditionally emphasized the link between the industry practice and the academic research. This has led to the development of normative models to help firms make decisions in a wide range of areas (e.g., capacity management, supply chain management, inventory management, dynamic pricing, and assortment planning).

The recent OM interest in sophisticated choice-based models of demand is aligned with this application focus, rooted in the need of high quality inputs for three key operational decisions: inventory, assortment, and prices. This motivation is different from marketing and econometrics, where estimating a choice model is an interesting goal unto itself given that it is frequently the case that a researcher wishes to uncover the underlying product features and consumer characteristics that influence the choice process.

While more than twenty years ago the academic research and the industry practices in revenue management (RM) and retail operations focused almost exclusively on those three key operational decisions through the development of sophisticated models and algorithms, the demand models that served as inputs were centered around *products*, leading to the so-called *independent demand paradigm*[1]. Under this paradigm, every product has its own stream of demand, independently of other products on offer. A dramatic change occurred when it was acknowledged that *customers* could substitute when their first alternative is not available. Customers create product demand in the first place, and understanding the drivers of their decision making is a key for building better models of demand, and hence, making better operational decisions.

Several studies have validated empirically the significant leverage that is obtained in the OM practice when accounting for choice-based models of demand. Yunes et al. [1] describe how they use conjoint survey results at John Deere & Co. to come up with consumer preferences based on the part-worth utilities for two of its product lines. For the two lines, a potential increase in profit from 8% to 18% has been identified, possible through reducing the number of configurations by 20% to 50% from present levels, while maintaining the current high customer service levels. In retail operations, Lee et al. [2] conduct a study of choice-based demand estimation at a university textbook retailer, and evaluate the impact of using the demand estimates to determine the inventory policy. They implemented the proposed model at the bookstore in a controlled field experiment and obtain over 10% increase in profit. In RM settings, the article by Vulcano et al. [3]

[1] A brief description of the independent demand model in Section 2.2 in the Appendix.

reports the results of a research study conducted in collaboration with a major U.S. commercial airline on origin-destination (O-D) markets between the New York metropolitan area and various airports in Florida. The authors calibrated a *multinomial logit* (MNL) model of demand using covariates such as price, arrival time, and departure dates from readily available airline data. Next, they used the demand estimates in a simulation study to assess the revenue performance of capacity controls computed under choice-based demand versus the incumbent EMSR-b (expected marginal seat revenue, version b) controls calculated based on the independent demand assumption, reporting average revenue gains between 1.4% and 5.3%. More recently, Dai et al. [4] also report significant revenue gains when accounting for choice behavior in a highly competitive O-D airline market in China.

The building block for estimating customer choice is the specification of a choice model, either parametric or nonparametric. Most of the proposals in the OM-related literature have been on the former for both estimation and assortment optimization (e.g., Musalem et al. [5], Kök and Fisher [6]). In this context, parametric means that the number of parameters that describe the family of underlying distributions is fixed and independent of the training data set volume. The greatest advantage of parametric models is the ability to include covariates, such as product features and price, that can help explain consumer preferences for alternatives. This also enables parametric models like the MNL or the *nested logit* (NL) with linear-in-parameters utilities to extrapolate choice predictions to new alternatives that have not been observed in historical data, and to predict how changes in product attributes such as price may affect choice outcomes. Yet, the drawback of any parametric model is that one must make assumptions about the structure of preferences and the relevant covariates that influence it, which requires expert judgment and trial-and-error testing to determine an appropriate specification.

Recently, with the rise of business analytics and data-driven approaches, there has been a growing interest in revenue predictions and demand estimation derived from nonparametric choice models. These models do not allow to make extrapolations and predict a change in the demand pattern due to a change in a product attribute, but are also built upon no assumption of the data structure and therefore they are only shaped by data and not by a superimposed choice pattern.

It is highlighted here that the computational efficiency of both estimation and assortment optimization procedures are key concerns for the performance of assortment planning solutions in retailing, where the number of SKUs in a family of products could typically be in the order of dozens to several hundreds. For instance, Oracle Retail Global Business Unit's tier-1 retailers[2] average 1,000 stores, with roughly 200 categories each containing 50

[2] These are retailers with annual revenues exceeding $5 billion and more than 250 stores.

to 600 items. (e.g., see Cohen at al. [7]). Similarly, in the context of a RM airline system, the usual approach is to subdivide the data from a large carrier network into finer-grained subsets (based on point of sale, time to departure, etc.), so airlines are really solving hundreds of thousands of different origin-destination (O-D) problems, which typically take large servers and several hours to run over nightly batch processes. For some discrete choice models such as the MNL and the Markov chain model, the assortment optimization problem can be solved efficiently, whereas for other discrete choice models such as the mixed MNL, it is NP-hard. This difference in terms of computational efficiency, can sometimes induce the modeler to select a choice model that is not the most accurate in terms of prediction but that can be handled properly in terms of solving the associated assortment problem within a reasonable time frame.

In summary, selecting a proper choice model for a particular operational context poses some concurrent challenges in terms of suitability of the model specification, prediction accuracy, (actual) revenue performance, and computational feasibility of the related estimation, and assortment and pricing optimization problems. This work intends to shed some light on the market conditions that favor some models over the other ones.

1.1 Contributions

The literature of discrete choice models used in operational contexts, together with the development of new estimation and assortment optimization procedures, has significantly expanded over the last two decades. The main focus of this literature stream and related industry practice has been on different variants of *random utility models*. Typically, most of the choice models evaluated or introduced are compared to the standard MNL model; they are rarely benchmarked to more sophisticated models in order to make a better judgment on their relative performance.

The main contribution of this work is to provide a systematic and exhaustive, comparative empirical study of the most important discrete choice models used in operational applications, which overall took over a year of CPU time. In particular, nine choice models are evaluated: MNL, exponomial, latent class MNL, nested logit, mixed logit, Markov chain (base case and two variants of it), and rank list-based. First, under a synthetic data environment, their performance on 3,600 instances is evaluated in terms of out-of-sample demand predictions by varying different features of the underlying data generation process such as the amount of training data, the degree of vertical differentiation between the products, the consistency of the consumer preferences, and the size of the offer sets. Next, the nine models are tested in terms of both demand and revenue predictions under two real datasets: (i) 3,500 bookings from five different hotels, and (ii) preferences from

5,000 individuals over different sushi types. Then, an experiment based on transactions obtained via synthetic preferences where a fraction of the customers are not fully rational is performed, in the sense that their behavior is not consistent with the utility maximization principle. Finally, in the appendix, the models were tested using as ground truths the standard latent class MNL model, and a nested logit model with varying prices. The variety of data sources and formats used (synthetic, semi-synthetic and real) provides another leverage in the contribution of this work to the literature.

These numerical results enable us to compare the prediction quality of the nine different choice models and two standard estimation methods (maximum likelihood and least squares), and allow us to characterize operational environments that seem more suitable for different choice models. The implementation of the estimation procedures are state-of-the-art methods, recently reported in the literature.

In a nutshell, two models stand out in terms of prediction accuracy and revenue performance: the exponomial model in small training data environments, and the Markov chain in large training data environments. This observation extends to the three types of instances evaluated: synthetic, semi-synthetic and real. Other general observations about the benchmarked models follow:

- The Markov chain model shows the most dramatic improvement as the volume of training data increases. In particular, it beats the latent-class MNL and the mixed logit when the ground-truth is the general stochastic preference model and even when it is the more structured latent-class MNL model. It also shows the best performance in the experiments using real data from sushi preferences and the best in 4 out of 5 hotel datasets tested.

- The predictive performance of all choice models deteriorates with offer sets that are more open (i.e., with fewer stockouts). In these settings, there is less information about substitution patterns to train models since most of the customers get their most preferable product. In this situation, very large volume of data would be needed for a choice model to leverage the performance of the traditional independent demand model.

- All choice models except for the ranked-list improve their predictive performance when synthetic ground-truths are generated from less-sparse models.[3] This is true when, in the interest of fulfilling a computational budget, only a relatively small number of preference lists are considered with nonzero probability.

[3] *Less-sparse* models is referred to models that are described by *many* customer types, where a customer type can be characterized by a total order of the products under consideration. This observation is further discussed in Section 5.2.3.

Besides assessing the quality of the predictions, the computational time required to estimate the different models in all the experiments was also measured. Some practical observations are provided below:

- The MNL was by far the fastest model to estimate.
- The nested logit and the exponomial were, on average, around 10 times slower than the MNL.
- The other models (Markov, ranked-list, latent class MNL, and mixed logit) were, on average, at least 100 times slower to estimate than MNL. This gap is expected to increase when larger datasets are used.
- Maximizing the log-likelihood function and minimizing least squares have very similar performance in terms of out-of-sample prediction accuracy for all the models tested.

The contribution spans two additional dimensions. First, all the estimation codes (in Python 3) and datasets are shared as part of the e-companion of this work, so that other researchers can replicate the results and use them as benchmarks for further studies. The files include a comprehensive documentation and the codes are organized in highly modular procedures that can be edited as needed to implement improvements and variants of the algorithms [4]. Second, an up-to-date survey of the literature on the choice models studied here is provided in section 2.2, including a summary of the state-of-the-art knowledge about the complexity of the related estimation and assortment optimization problems (see Section 2.3 therein).

The remaining content is organized as follows. In Section 2 the basics of discrete choice models, the estimation procedures and the performance metrics used is described. Sections 3 and 4 provide details on the development of the Python code and the infrastructure used to run the experiments. Sections 5 and 6 are the core of this work, where the results based on synthetic and real data are respectively analyzed. Section 7 evaluates the performance of the nine choice models when consumers display boundedly rational choice behavior. Finally, concluding remarks are discussed in Section 8.

[4] The estimation codes and the instance sets are available at `https://github.com/ajgara/choice-models`.

2. DISCRETE CHOICE BASICS

This section is started with some basic definitions and concepts on discrete choice theory, followed by a description of the implemented estimation techniques and the performance metrics used.

2.1 Random utility models

Let $\mathcal{N} = \{1, 2, \ldots, n\}$ denote a universe of substitutable products (e.g., coffee or milk in a grocery store) and let product 0 denote the "outside" or "no-purchase" option.

A *discrete choice model* describes the decision makers' choices among a set of alternatives. Formally, a discrete choice model is a function $\mathcal{P} : \mathcal{N} \cup \{0\} \times 2^{\mathcal{N}} \to [0, 1]$ such that $\mathcal{P}(j|S)$ denotes the probability that an individual would select alternative $j \in \mathcal{N} \cup \{0\}$ given that the subset of products S was offered.[1]

A key concept in discrete choice theory is the notion of *random utility*. Under random utility models (RUMs), the decision maker (i.e., customer) would obtain certain level of utility from each of the alternatives in $\mathcal{N}$ and the no-purchase option: $U_1, \ldots, U_n$, and U_0. The consumer utility for product j is typically decomposed as $U_j = v_j + \varepsilon_j$, where v_j represents the nominal utility, and ε_j captures the factors that affect utility but are non observable. The modeler observes the different product attributes that influence the nominal utility v_j. Typically, that influence is modeled with a linear function, i.e., $v_j = \beta_j + \boldsymbol{\beta}^T \boldsymbol{x}_j$, where $\boldsymbol{x}_j$ is a vector of observable attributes for alternative j, and $\boldsymbol{\beta}$ is a vector of coefficients that assign a weight to each attribute plus an alternative specific constant β_j.[2] On the contrary, the modeler does not know ε_j and therefore treats these terms as random variables. Indeed, different assumptions related to the specification of the random terms would lead to different choice models. When the decision maker is confronted with a choice instance, given a particular realization of the random term, she ranks the utilities and chooses the top alternative out of all those being offered.

A fundamental property of RUMs is that "only differences in utilities of different options matter" (e.g., see Train [8]). To account for this, the usual approach is to normalize one of the alternative specific constants to zero. In order to diminish the impact of the

[1] The more appropriate functional notation would be $\mathcal{P}(j, S)$, but the common practice is to write $\mathcal{P}(j|S)$.

[2] It is noted that this modeling approach ignores individual-specific attributes by solely focusing on product attributes. Thus, customer differences are treated as latent, which is a standard practice in OM and differs from the usual models in marketing.

attribute selection process in the quality of the estimations and center the discussion around the inherent quality of the choice models, in most of the study this step is bypassed and the experiments focus on the nominal utilities v_j.[3] From here onwards, it is assumed w.l.o.g. that the no-purchase option has a nominal utility of zero (i.e. $v_0 = \beta_0 = 0$).

In this work nine variants of random utility models extensively used in the OM-related literature were tested. They are listed here jointly with the shorthand used in the companion figures and tables: multinomial logit (mnl), exponomial (exp), latent class MNL (lc), nested logit (nl), mixed multinomial logit (mx), Markov chain (mkv), reduced Markov chain (mkvr), Markov chain of rank 2 (mkv2), and rank list-based (rl). For completeness, as it was mentioned above, a survey of these models is included in Section 2.2. For the Markov chain model, three different versions are considered: one including all possible transitions, one where the transition matrix has rank 2, and one where the nontrivial transitions are designed following a vertical differentiation of the products.

This section is ended by noting that, while most choice models in the literature belong to the random utility family, there exists a new stream that focuses on models that go beyond the RUM class. These models try to capture well documented behavior that is (or at least appears to be) sub-optimal or boundedly rational, such as violations of *regularity* (Berbeglia and Joret [9]) or the *decoy effect* (Huber et al. [10]). Examples of these models are the perception adjusted Luce model (Echenique et al. [11]), the generalized stochastic preference (GSP) (Berbeglia [12]) and the general Luce (Echenique and Saito [13]). In this work results from experiments where the ground truth is defined by the GSP model is reported (in Section 7).

2.2 A survey of the discrete choice models benchmarked

A description of the discrete choice models benchmarked in this study, including related references and documented theoretical results proceeds. In the preliminary studies two additional naïve models were also included: independent demand (discussed in Section 1) and random (where any item in the offer set could be selected equally likely). Since the nine models presented in the numerical experiments dominate these two, in the interest of the clarity of the figures and tables, there are not included in the results. This section is closed with a summary table accounting for the most relevant model features.[4]

[3] For completeness, one set of experiments accounting for product prices is included in Appendix A5.

[4] Strauss et al. [14] independently developed a review of choice models used in revenue management contexts which includes a short description of some of the models presented here.

2.2.1 Independent demand

As its name suggests, the probability that a consumer selects a product $i \in \mathcal{N}$ under this choice model is *independent* of the offer set. Formally, a discrete choice model in this family must satisfy that

$$\mathcal{P}(i|S) = \mathcal{P}(i|S') \text{ for every } i \in \mathcal{N}, \text{ and } S, S' \subseteq \mathcal{N} \text{ , with } i \in S \cap S'.$$

This model can therefore be specified by a discrete probability distribution $x_1, \ldots, x_n, x_0$, over each of the products, including the no-purchase option. The weight of the choice in this distribution can be seen as its market share. When the subset $S \subseteq \mathcal{N}$ is offered and a customer arrives intending to buy product $j \in S$, she buys it and leaves. Otherwise, if $j \notin S$, she quits without purchasing. Then, we have:

$$\mathcal{P}(j|S) = x_j \text{ when } j \in S, \text{ and } \mathcal{P}(0|S) = x_0 + \sum_{i \in \mathcal{N} \setminus S} x_i.$$

The maximization of the log-likelihood function has to account for the constraints:

$$\sum_{i=0}^{n} x_i = 1; \quad x_i \geq 0, \forall i.$$

For the purpose of this works ML estimation, the formulation to optimize is a concave maximization problem subject to a linear constraint, which can be solved efficiently with standard nonlinear optimization algorithms.

This is the simplest demand model that has been traditionally around in RM and retail operations, and assumes no substitution behavior (or, more precisely, substitution only onto the no-purchase option). As the demand lost when a customer's first choice is unavailable (referred to as spilled demand) is ignored, the resulting demand forecasts might be negatively biased. Thus, the first leverage to the independent demand model in the context of time series forecasts was the use of "unconstraining" methods to extrapolate the underlying demand distribution parameters from truncated demand data collected over previous selling opportunities (e.g., see Queenan et al. [15]).

2.2.2 Multinomial logit

The multinomial logit (MNL), first introduced by Luce [16], is one of the most studied discrete choice models and one of the most frequently used in practice.

Under the MNL model, the random noise ϵ_j in the utility of product j, U_j, follows a standard Gumbel distribution with location parameter zero. The probability that an individual will select alternative $j \in S \cup \{0\}$ is

$$\mathcal{P}(j|S) = \frac{\exp(v_j)}{\sum_{i \in S \cup \{0\}} \exp(v_i)}$$

where $\exp(v_0) = 1$.[5] The associated log-likelihood function is concave and can be efficiently optimized using standard nonlinear optimization packages.

A convenient property of the MNL model is that the likelihood of purchasing a given product can be readily recalculated if the availability of the products changes. However, the MNL has significant restrictions in terms of modeling choice behavior, most notably the property of independence from irrelevant alternatives (IIA). Briefly, this property says that the ratio of purchase probabilities between two available alternatives remains constant regardless of the choice set containing them, which could lead to unusual substitution patterns. If the choice set contains alternatives that can be grouped such that intragroup alternatives are more similar than intergroup alternatives, the MNL model is not realistic because adding new alternatives reduces the probability of choosing similar alternatives more than dissimilar alternatives. The reader is referred to the discussion on the famous red-bus-blue-bus paradox in Train [8].

Despite the IIA limitations, several successful implementations of the MNL model in retail and RM settings have been reported (e.g., Guadagni and Little [17], and more recently, Ratliff et al. [18], Newman et al. [19]), suggesting that it could provide an interesting balance between specification and estimation errors.

Regarding the associated assortment optimization problem (i.e., when prices are fixed and the decision is about which products to include in the offer set), Talluri and van Ryzin [20] showed that it is computationally tractable since the MNL satisfies the general conditions of *revenue-ordered assortments*; i.e., given a set of n products with prices $r_1 \geq r_2 \geq \cdots \geq r_n$, the assortment that maximizes expected revenues is given by a complete set of the highest-priced products $A_k = \{1, 2, \ldots, k\}$, so that the algorithm is simply about checking the expected revenues coming from n complete sets $A_1, \ldots, A_n$, and retaining the one that achieves the highest value.[6]

The price optimization problem for a given set of products under the MNL model with identical price-sensitivity parameters was studied by Anderson and de Palma [22], and Hopp and Xu [23], who show that the profit of the firm is maximized when the firm uses a constant markup for all products.

2.2.3 Exponomial

A parametric discrete choice model that recently caught some attention in the OM literature is the exponomial choice (EC) model proposed by Daganzo [24], and revisited by

[5] In the particular case where all v_j take the same value, the model reduces to the random model referred to at the beginning of the section.

[6] The result was later extended by Li [21] to the case of profit maximization under unequal cost parameters.

Alptekinoğlu and Semple [25], which assumes a negatively skewed distribution of consumer utilities. This skew can be justified when consumers would not choose to overpay if they were well informed about products and prices.

The exponomial discrete choice model is another example of RUM. Under this model, each product j has utility $U_j = v_j - z_j$, where v_j is the maximum utility that any consumer could theoretically assign to product j, and $z_j \geq 0$ is an exponentially distributed random variable that discounts the maximum possible utility accounting for the unobserved personal preferences of different consumers in the population.

The choice probabilities in this model can be derived in closed-form as an exponomial (a linear function of exponential terms). Knowing that a customer always chooses the option that maximizes her perceived utility, we have for $j \in S \cup \{0\}$ (as seen in Alptekinoğlu and Semple [25]):

$$\mathcal{P}(j|S) = \frac{\exp(-\sum_{i=j}^{|S|}(v_i - v_j))}{|S| - j + 1} - \sum_{k=0}^{j-1} \frac{\exp(-\sum_{i=k}^{|S|}(v_i - v_k))}{(|S| - k)(|S| - k + 1)},$$

where it is assumed w.l.o.g. that the options in $S \cup \{0\}$ are ordered and relabeled so that $v_0 \leq v_1 \leq v_2 \leq ... \leq v_{|S|}$.

Following Alptekinoğlu and Semple [25], the log-likelihood function is concave and can be easily optimized with standard nonlinear optimization packages. However, the price optimization problem is more complex than under MNL -where products have a constant mark-up. Here, the optimal markups increase in expected utilities for all products that are more desirable than the outside option even for the case of constant price sensitivity across products. Recently Aouad et al. [26] proposed a FPTAS for the assortment problem when prices are fixed. However, the question of whether this problem is NP-hard remains open.

2.2.4 Latent class MNL

The latent class MNL (LC-MNL) is a flexible model that can be interpreted as a weighted average MNL, with K classes, class-dependent utilities, and a market composition described by proportions $\lambda_1, \ldots, \lambda_K$, that capture heterogeneity in taste. The probability that product j is chosen when assortment S is offered, is given by:

$$\mathcal{P}(j|S) = \sum_{k=1}^{K} \lambda_k \frac{\exp(v_{j,k})}{\sum_{i \in S \cup \{0\}} \exp(v_{i,k})},$$

where $v_{j,k}$ is the nominal utility that a customer of segment k gives to product j, and where $v_{0,k} = 0$, for all k. The proportions satisfy: $\lambda_k \geq 0$, $\sum_{k=1}^{K} \lambda_k = 1$.

The LC-MNL model is, in fact, a very flexible model to accommodate different possible substitution patterns, avoiding the aforementioned IIA limitation of the MNL.

On the downside, the associated log-likelihood function does not have much structure and hence is challenging to optimize. A promising approach has been developed by Jagabathula et al. [27], formulating the problem as a constrained convex program and using a conditional gradient method to optimize it. The overall idea of the method is to incrementally add (latent) classes as long as the log-likelihood ratio (i.e., the ratio between the log-likelihoods of two consecutive steps in the iterative process) be smaller than $1e-7$, or a time limit of 30 minutes. This method was used to report the results in the main section of this work, due to its good time performance.

Liu and van Ryzin [28] extend the assortment problem for the single class MNL model to the case of customer segments with disjoint consideration sets, where the sets of products of interest for different segments are non-overlapping. They prove that the assortment problem still remains computationally tractable. However, when the segments are overlapping, Miranda Bront et al. [29] show that the assortment problem becomes NP-Hard. These authors also formulate the problem as a mixed integer program as part of a column generation procedure to solve a large scale network RM problem.

In a recent paper, Feldman and Topaloglu [30] develop a decomposition approach to construct an upper bound on the optimal expected revenue, and show numerically that the bound is indeed tight. For the particular case when there are K possible realizations of the choice model parameters, Rusmevichientong et al. [31] show that revenue-ordered assortments provide an approximation guarantee of $\max\{1/K, \frac{1}{\lceil n/2 \rceil}\}$ and this bound is tight. Recently, Berbeglia and Joret [9] showed that revenue-ordered assortments provide a revenue guarantee of a $\max\{1/h, \frac{1}{1+\ln(r_{max}/r_{min})}\}$ fraction of the optimal revenue where h, r_{max} and r_{min} denote the number of different prices among all products, the maximum price, and the minimum price respectively. This bound is also tight and applies to any choice model that satisfies regularity.

2.2.5 Nested logit

The nested logit model partitions the set of products (including the no-purchase option) into K subsets or "nests", denoted $\mathcal{N}_1, \ldots, \mathcal{N}_K$. The utility of product j is still $U_j = v_j + \varepsilon_j$. The random noises ε_js, with associated vector $< \varepsilon_0, \ldots, \varepsilon_n >$, have joint cumulative distribution:

$$\exp(-\sum_{k=1}^{K}(\sum_{j\in\mathcal{N}_k}\exp(-\frac{\varepsilon_j}{\lambda_k}))^{\lambda_k})$$

As noted in Train [8], this distribution is a type of general extreme value (GEV) distribution, a generalization of the Gumbel distribution of the MNL model. The parameter λ_k is a measure of the degree of independence of unobserved utilities among the alternatives in nest $\mathcal{N}_k$. Higher values of λ_k indicate more dissimilarity for the alternatives within $\mathcal{N}_k$. In order to keep the model within the unbrella of RUMs, $\lambda_k < 1$ is required, for all k (see Davis et al. [32]).

The associated choice probability for product $j \in \mathcal{N}_k \cap (S \cup \{0\})$ can be expressed as:

$$\mathcal{P}(j|S) = \frac{\exp(\frac{v_j}{\lambda_k})(\sum_{i \in \mathcal{N}_k \cap (S \cup \{0\})} \exp(\frac{v_i}{\lambda_k}))^{\lambda_k - 1}}{\sum_{\ell=1}^{K} (\sum_{i \in \mathcal{N}_\ell \cap (S \cup \{0\})} \exp(\frac{v_i}{\lambda_\ell}))^{\lambda_\ell}}.$$

The nested logit model alleviates the issue of IIA exhibited by the MNL model, though it still inherits the IIA property within a nest so that when an alternative is removed from it, the probability of the other choices in that nest must increase proportionally in the same amount. On the downside, it is often challenging to partition the products into different nests, and the model is limited because it requires each product to belong to exactly one nest. In this works implementation, the products are partitioned into three nests: one nest includes only the no-purchase alternative, and products $1, 2, \ldots, n$ are split evenly between the other two nests. To estimate the parameters of this model direct optimization of the log-likelihood function was used, using standard nonlinear optimization packages, subject to the constraints $\lambda_k < 1$, for all k.

The log-likelihood function does not have much structure and is challenging to optimize. To overcome this, other approaches have been proposed. For instance, Brownstone and Small [33] describe a two-stage estimation procedure in which the (non-separable) log-likelihood function is split in two terms which are optimized sequentially. Different from ML estimators, these estimators are not efficient. Furthermore, the covariance matrix has to be computed separately, and the estimates are not even guaranteed to ensure consistency with utility maximization. In his widely cited book, Train [8, Section 4.2.4] recommends to use the simultaneous optimization of the ML approach over the two-stage procedure. Following this, and in order to be consistent with the ML method used for the other models, the log-likelihood function was directly optimized in the experiments.

The assortment optimization problem scales its complexity from the MNL model to the NL model. Davis et al. [32] showed that the problem is polynomially solvable when the nest dissimilarity parameters of the choice model are less than 1 and the customers always make a purchase within the selected nest. Relaxing either of these assumptions makes the problem NP-hard. To deal with the NP-hard cases, they develop parsimonious collections of candidate assortments with worst-case performance guarantees and formulate a convex program whose optimal objective value is an upper bound on the optimal expected

revenue. Gallego and Topaloglu [34] studied the same problem under cardinality and space constraints on the offered assortment, which limit the number of products and the total space consumption of the products offered in each nest, respectively. They show that the optimal assortment under cardinality constraints can be obtained efficiently by solving a linear program, but the assortment optimization problem under space constraints is NP-hard.

A particular case of the general NL model is the d-level NL model, where nests are organized as trees of depth d. Li et al. [35] provide a polynomial algorithm to find the optimal assortment. For the price optimization problem when the assortment of offered products is fixed, the expected revenue function is not concave in the product prices. Yet, they develop an iterative algorithm that generates a sequence of prices converging to a stationary point. Their numerical experiments show that the proposed method converges much faster than gradient-based methods.

2.2.6 Mixed logit

The mixed MNL (MMNL) model accounts for random taste variation, unrestricted substitution patterns, and correlation in unobserved factors over time. The market is described by a heterogeneous composition of segments, each of them described by an MNL model. In this regard, it becomes a generalization of the LC-MNL where the mixing distribution of classes is continuous.

McFadden and Train [36] establish the important result that, in theory, any random utility model can be captured by a correct specification of the mixing distribution here, providing strong support to the flexibility of this model in terms of its capability to represent more general customer choice behavior.

The probability of choosing j given an offer set $S \subseteq \mathcal{N}$ can be expressed as:

$$\mathcal{P}(j|S) = \int \frac{e^{v_j}}{\sum_{i \in S \cup \{0\}} e^{v_i}} f(v|\theta) dv.$$

The attraction vectors v_js are weighted by a density function f, so this probability density function (pdf) must be specified to fully determine this model. In this work f was chosen so that $f(x|\mu, \Sigma)$ describes the density of a multivariate normal distribution $N(\mu, \Sigma)$ with mean μ and diagonal covariance matrix Σ . In particular, when choosing a multivariate normal distribution the model ends up being equivalent to a probit model simulated through a smoothed accept-reject method (see Train [8]).

In this case, the parameters to be estimated are μ and the entries in the diagonal of the matrix Σ. As this integral must be solved numerically through simulation, the estimation of the parameters is generally computationally intensive.

2.2.7 Markov chain

Under the Markov chain (MC) model proposed by Blanchet et al. [37], individual preferences are built using a Markov chain in which states are the alternatives or products (including the no-purchase).

The building block for this model is a bit different from others discussed before. Instead of using a utility-based approach to modeling substitutions that require specifications of utility functions, the MC takes substitutions directly as a primitive. With probability p_j an individual has alternative j as her most favorite. If alternative j is not available, with probability p_{ji} the individual who preferred j jumps to select alternative i. If, in turn, alternative i is not available, with probability p_{ik} she jumps to alternative k, and so on. Because it is assumed that for all $j \in \mathcal{N}$, $\sum_{i \in \mathcal{N} \setminus \{0,j\}} p_{ji} < 1$, this process will finish either when the consumer jumps to the no-purchase option (in which case selects nothing) or when she arrives to an alternative from the set S. The probabilities $\mathcal{P}(j|S)$, for all $j \in S \cup \{0\}$, must satisfy the following set of linear balance equations:

$$\mathcal{P}(j|S) = p_j + \sum_{i \in \mathcal{N} \setminus (S \cup \{0\})} p_{ij} \mathcal{P}(i|S).$$

A priori, this model seems unrelated to the class of RUMs and is motivated from a completely different perspective. However, Blanchet et al [37] show that it provides a simultaneous approximation for all random utility based discrete choice models including the multinomial logit (MNL), the nested logit and the latent class MNL. Furthermore, they show that the MC model provides a good approximation in choice probabilities for any random utility model under some mild assumptions. More recently, Berbeglia [38] showed that the MC effectively belongs to the RUM family.

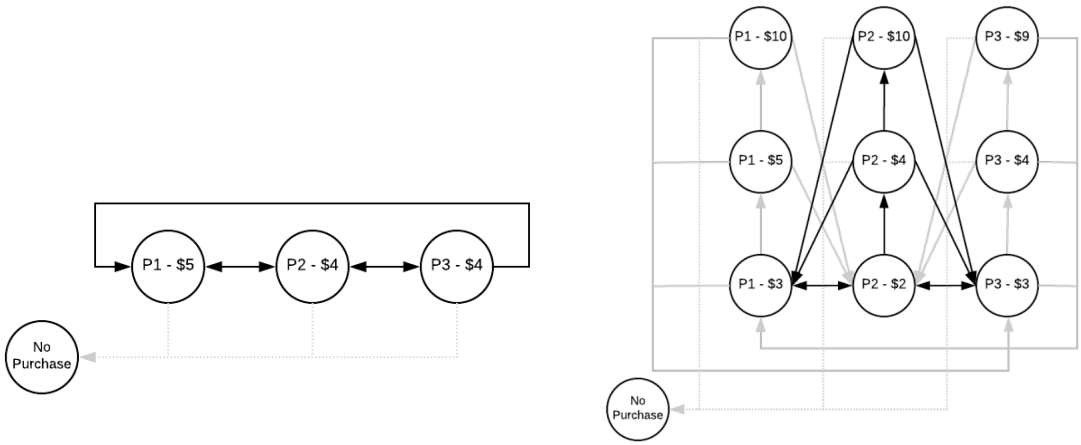
Blanchet et al [37] also show that the underlying parameters of the MC model are identifiable from the choice probabilities for the full assortment, together with choice probabilities for assortments of all-but-one products. However, it may be unrealistic or unprofitable to offer assortments of such large cardinality. Gupta and Hsu [39] show that the parameters may be identified through a simple and efficient parameter recovery algorithm, from assortments of sizes two and three, regardless of the total number of products.

Blanchet et al [37] further show that the MC-related assortment optimization problem can be solved in polynomial time. through a reduction to an optimal stopping problem. They present an iterative algorithm that can solve the problem efficiently at scale in the uncapacitated assortment case. Feldman and Topaloglu [40] cast this assortment problem into a linear programming perspective: They relate the probabilities of purchasing different products under the MC model to the extreme points of a polyhedron and use

this connection to formulate a linear program that can be used to obtain the optimal set of products to offer. More recently, Desir et al. [41] provide constant factor approximation algorithms for the cardinality- and capacity-constrained assortment-optimization problems under the Markov chain model.

Yet, the practical challenge of the Markov chain model seems to rest on its estimation. The EM-based proposal of Simsek and Topaloglu [42], though computationally intensive, constitutes a step in this direction. The reader is referred the reader to Appendix 2 where a couple of plots comparing the direct optimization of the log-likelihood function and the EM algorithm show the better performance of the latter. Given this evidence and its faster computational time, the EM algorithm was used for the experiments. In each iteration the estimation step consists of solving a linear system of equations and the maximization step is solved via a closed form expression.

Observe that while other parametric discrete choice models presented in this work have a number of parameters that is proportional to the number of products, the Markov chain model has $O(n^2)$ parameters. This large number of parameters can trigger over-fitting problems in the estimation phase when there is little historical data. In order to overcome this issue, two variants of the Markov chain model that reduce the number of variables to $O(n)$ were also implemented. The first variant is labelled as the *reduced Markov chain model*. Here, at most three non-zero outgoing substitutions from each product are allowed (besides the substitution to the no-purchase option). Figure 2.1 illustrates non-null substitution probabilities of a random instance and a structured instance with only three product types. Note that both schemes correspond to Markov chains with a single absorbing "no-purchase" state, and a single transient communication class including all nodes representing products.



(a) Random instance generator (b) Structured instance generator

Fig. 2.1: Sketch of the reduced Markov chain model used in (a) random instances, and (b) structured instances. The example represents only three products, but the actual implementations were for 10 products (case (a)), and 5 product types with 3 prices (case (b)). For the latter, the structure of the outgoing arcs is based on the vertical differentiation of the products.

The second variant reduces the number of parameters by restricting the transition matrix $M_{ij} = p_{ij}$ to have rank two. This is implemented by using four vectors such that $M = u^t v + w^t z$, which ends up summing a total of $O(5n)$ parameters. This variant is labeled *Markov chain of rank 2*.

2.2.8 Rank list-based

Rank list-based (or *stochastic preference*) choice models of demand have recently received particular attention in the retail operations and revenue management literature. This type of nonparametric model has previously been studied in economics and mathematical psychology (see Block and Marschak [43], Manski [44], Falmagne [45], Barbera and Pattanaik [46]). One of the first applications in OM settings was the retail assortment problem studied by Mahajan and van Ryzin [47].

In this model, individuals are assumed to draw a rank-based preference for the products in each store visit. That is, each individual has a preference list (or total ranking), σ, of the products in $\mathcal{N}$, which is independent of the offer set S that she faces in a particular store visit. Since each visit is treated independently and anonymously, different store visits from the same customer could imply the draw of different rankings.

A customer represented by σ in a store visit prefers product j to product h if and only if $\sigma(j) < \sigma(h)$, with the most preferred product being $\sigma(1)$. This preference extends

to the no-purchase alternative as well, since the ranking includes alternative 0. Note that the meaningful rankings are only the ones truncated at the position of product 0; that is, a preference list σ will have as many elements as $\sigma(0)$. A *stochastic preference model* for the universe of alternatives $\mathcal{N}$ is defined by a probability distribution $\mathbb{P}$ over the set of all rankings σ of the elements in $\mathcal{N}$, so that $x_i = \mathbb{P}(\sigma_i)$, $i = 1, \ldots, K$, with $K = O(n!)$. In a given store visit, if there is an arrival with a draw σ, that customer chooses $j = \arg\min_{j \in S \cup \{0\}} \sigma(j)$.

For the full set of rankings $\{\sigma^{(1)}, \ldots, \sigma^{(K)}\}$ and $j \in S \cup \{0\}$, define the set:

$$\mathcal{M}(j, S) = \{i : \sigma^{(i)}(j) < \sigma^{(i)}(k), \forall k \in S \cup \{0\}, k \neq j\}.$$

In words, $\mathcal{M}(j, S)$ is the set of ranking labels for which product j is ranked highest among the available alternatives in $S \cup \{0\}$. Given a probability distribution $\boldsymbol{x}$ over the set of rankings, the probability that a random arrival chooses product j when set S is offered, is given by

$$\mathcal{P}(j|S) = \sum_{i \in \mathcal{M}(j,S)} x_i, \quad \text{if } j \in S \cup \{0\},$$

and $\mathcal{P}(j|S) = 0$ if $j \notin S \cup \{0\}$.

Block and Marschak [43] studied the relationship between stochastic preferences and discrete choice models based on random utilities, showing that a system of choice probabilities $\mathcal{P}$ is a RUM if and only if $\mathcal{P}$ is induced by some stochastic preference model. As a consequence, any RUM model can be specified through a utility-based model or through a stochastic preference model (i.e., as a pdf over rankings).

Applications of rank-based models in the context of retail operations include Smith et al. [48], Honhon et al. [49, 50], Jagabathula and Rusmevichientong [51], Jagabathula [52], Bertsimas and Mišić [53], and Jena et al. [54]. In the context of airline revenue management, Zhang and Cooper [55], Chen and Homem-de-Mello [56], Chaneton and Vulcano [57], and Kunnumkal [58] also use this rank-based choice model.

One of the first papers to address the estimation of such model was Farias et al. [59], who use a robust approach, finding the distribution over rankings that produces the worst-case revenue compatible with the observed data for a given fixed assortment. These robust revenues are computed approximately using constraint sampling. They then show that the demand model constructed by the robust approach is approximately the *sparsest-choice model*, where sparsity is measured by the number of rankings that occur with positive probability in the population, and that the sparsity grows with the amount of observed data. For the same demand model, a column generation-based procedure to compute maximum likelihood estimates for this model based on duality arguments is developed in van Ryzin and Vulcano [60]. In this work the latter algorithm was used

to estimate this model. Following the guidelines of that paper, the iterative process was started from the independent demand model; that is, from a collection of lists given by singletons, with one singleton per product. The master program (or "restricted estimation problem") was solved by using the expectation-maximization (EM) procedure of van Ryzin and Vulcano [61]. For a given set of lists, the master procedure stops if either during five consecutive EM iterations the difference of the log-likelihood values remains smaller than $1e-6$ or it reaches 5 minutes of computing time. For the stopping criteria of the column generation process the log-likelihood ratio was used (asking for the ratio between the log-likelihoods of two consecutive steps in the iterative process to be smaller than $1e-6$), or a time limit of 30 minutes.

One of the downsides of this model is its non-identifiability (i.e., two different pmfs may explain the same observed data; see Sher et at. [62, Theorem 5]). Here, the potential number of preference lists is $O(n!)$, whereas the potential number of offer sets from where to estimate the consumers' preferences is $O(2^n)$. Nevertheless, the evidence reported in the literature about its potential is quite promising: Farias et al. [59] used a dataset consisting of automobile sales transaction data from a major U.S. automaker, and demonstrated a 20% improvement in prediction accuracy over the MNL and MMNL models. They further assessed that the improvement can translate into a 10% increase in revenues from optimizing the offer set.

2.3 Summary of relevant model features

Model	**Number of parameters**	**Structure of log-likelihood function**	**Estimation**	**Assortment optimization time complexity**
Independent demand (id)	$O(n)$	Concave	Standard optimization methods	$O(n)$; Checking n sets consisting of the i most expensive products
MNL (mnl)	$O(n)$	Concave	Standard optimization methods	$O(n)$; Checking n sets consisting of the i most expensive products
LC-MNL (lc)	$O(nk+k)$	Non-linear, non-concave in general	EM algorithm; Conditional gradient algorithms	NP-Hard
Nested logit (nl)	$O(n+k)$	Non-linear, non-concave	Standard optimization methods	NP-Hard; Polynomial under certain conditions
Exponomial (exp)	$O(n)$	Concave	Standard optimization methods	Open question (a FPTAS exists)
Mixed MNL (mx)	$O(2n)$	Non-linear, non-concave in general	Numerical simulation	NP-hard
Rank based (rl)	$O(n!)$	Concave	Non-linear optimization by column generation	NP-hard
Markov chain (mkv)	$O(n^2)$ ($O(n)$ for reduced and rank 2)	No closed form expression; it involves solving multiple systems of linear equations	EM algorithm	Polynomial, solving a linear program

2.4 Estimation

Estimation is the problem of finding model descriptors that best explain a given set of observed data. In this section the most important methodologies for parameter estimation is reviewed and some of its properties are discussed. To unify terminology, the discussion will be focused on the estimation of *parameters*, though the same ideas apply to the estimation of the descriptors of nonparametric models (e.g., the empirical distribution of a demand function).

The *training* or *in-sample* dataset consists of a sequence of T observations: $\{(j_1, S_1), \ldots, (j_T, S_T)\}$, where S_t is the set of products offered in period t, and $j_t \in S_t \cup \{0\}$ is the product purchased in that period.

An *estimator* represents a systematic "guess" about the parameters of the underlying, ground truth distribution from which these T observations have been drawn. The estimator defines the rule for calculating an *estimate* of the parameter based on the sample. For j_t being the t-th observed transaction drawn from a distribution $\mathcal{P}(j_t|S_t, \boldsymbol{\beta})$ -where the dependence on $\boldsymbol{\beta}$ is explicited, the estimation problem is to determine the unknown parameters $\boldsymbol{\beta}$ using only the sample of T observations $\{(j_1, S_1), \ldots, (j_T, S_T)\}$.

This work focuses on two different estimation methods widely used in both the academic literature and the industry practice: maximum likelihood and least squares. The computational results showed that the performance of these estimation procedures is very similar, which is consistent with the results reported by a recent comparison carried out by Genschel and Meeker [63]. A description of least squares is deferred to the Appendix A1.

Maximum likelihood (ML) estimators are based on finding the parameters that maximize the probability of observing the sample data, where *likelihood* is defined as the probability of the full set of observations occurring. Under this method, given a choice model $\mathcal{M}$ with parameters $\boldsymbol{\beta}$, the likelihood of observing the set of independent and identically distributed (i.i.d.) observations is then

$$\mathcal{L} = \prod_{t=1}^{T} \mathcal{P}_{\mathcal{M}}(j_t|S_t, \boldsymbol{\beta}).$$

The ML estimation problem is to find a vector $\boldsymbol{\beta}$ that maximizes the likelihood function. Taking advantage of the monotonicity of the log function, the usual approach though is to maximize the log-likelihood function, i.e., to solve

$$\max_{\boldsymbol{\beta}} \sum_{t=1}^{T} \log \mathcal{P}_{\mathcal{M}}(j_t|S_t, \boldsymbol{\beta}). \tag{2.1}$$

In some special cases, this problem can be solved in closed form. Otherwise, if $\mathcal{P}(j_t|S_t, \boldsymbol{\beta})$ is differentiable in $\boldsymbol{\beta}$, then gradient-based optimization methods such as Newton's method

can be used to solve it numerically. Nevertheless, the log-likelihood function sometimes may not have much structure (e.g., may not even be quasiconcave) or involve numerous variables, and standard nonlinear optimization methods may become very computationally intensive and inefficient.

An alternative approach that has become popular in the context of demand estimation in settings affected by stockouts and substitution effects is the expectation-maximization (EM) algorithm (e.g., Anupindi et al. [64], Kök and Fisher [6], Conlon and Mortimer [65], Stefanescu [66], van Ryzin and Vulcano [61], Jagabathula and Rusmevichientong [67]). The EM method was proposed by Dempster et al. [68]. Its fundamental idea is to replace a potentially complex *incomplete data* log-likelihood function (i.e., the function (2.1) that emerges from the purely observed data) by the *complete data* log-likelihood function; that is, the likelihood function we would get if we were able to directly observe some hidden data. The procedure then works using alternating steps of: i) replacing the complete data with beliefs (expectations) about their values conditional on the observed data and current parameter estimates (the E-step), and ii) maximizing this function to obtain improved parameter estimates (the M-step). Its practical effectiveness depends on the computational efficiency of both steps. Even though it is well acknowledged that the convergence of the EM algorithm could be slow, particularly with large numbers of parameters or high degrees of data censoring, there have been several successful implementations reported in the literature for different choice-based demand models that position it as an appealing candidate to compute ML estimates.

It is also worth highlighting few desirable statistical properties of ML estimators. In cases where the underlying demand model is identifiable and some other mild conditions (e.g., see Green [69]), maximum likelihood estimators are consistent (i.e., they converge in probability to the true parameter values), asymptotically normal (i.e., $\sqrt{T}(\hat{\boldsymbol{\beta}}-\boldsymbol{\beta})$ converges in distribution to a multivariate normal distribution), and asymptotically efficient (i.e., asymptotically unbiased and attaining equality of the Cramer-Rao lower bound for the variance, asymptotically).

Before proceeding, it is highlighted that the estimation procedures studied (both least squares and ML) implement some kind of regularization which could in principle mitigate the overfitting risk of the high dimensional choice models. The strategy of the state-of-the-art methods that were implemented was to limit the number of nonzero parameters in the high dimensional models. In particular, for the latent class MNL, the criteria has been to add a new (latent) class as long as the log-likelihood increase is significant enough, as described in Jagabathula et al. [27]. The rank list-based model is estimated via the column generation algorithm proposed by van Ryzin and Vulcano [61], where a new column

is added as a new rank list with nonzero probability only when the increase in ratio is significant enough. For the Markov chain model, the EM procedure described by Şimşek and Topaloglu [42] was implemented, which does not incorporate an explicit regularization, but the Markov chain of rank 2, where the number of parameters to estimate is reduced from $O(n^2)$ to $O(5n)$, is a means to regularize the log-likelihood function. There are alternatives for implementing regularization such as penalizing nonzero coefficients in the log-likelihood function, but these are beyond the scope of this work and may be worth further studied.

2.5 Performance metrics

In this work two versions of the root mean square error (RMSE) are considered. The RMSE is a frequently used measure of the differences between sample values predicted by a model and the values actually observed, providing a sense of how far from the actual ground truth model the estimated model is. It is a common measure of accuracy throughout the choice-based OM literature.

Consider two different models: θ being the ground truth, and ϕ being an estimation. The (soft) root mean squared error is calculated as follows:

$$RMSE^{soft}(\mathcal{P}_\theta, \mathcal{P}_\phi) = \sqrt{\frac{\sum_{S\subseteq\mathcal{N}} \sum_{j\in S\cup\{0\}} (\mathcal{P}_\theta(j|S) - \mathcal{P}_\phi(j|S))^2}{\sum_{S\subseteq\mathcal{N}} (|S|+1)}}$$

The additive 1 in the denominator stands for the no-purchase alternative. Note that this metric is computed over all possible offer sets and all possible selections for each of the offer sets. Therefore, it does not actually need transactions (e.g. out-of-sample data) as it just compares two choice models directly over every possible (product, assortment) pair.

In scenarios were the ground truth is not known, one can approximate this measure by replacing ground truth probabilities $\mathcal{P}_\theta$ with an indicator function I that checks whether product j was purchased from offer set S in each of the out-of-sample transactions. This is known as the hard root mean squared error. Let T denote the set of out-of-sample transactions, then the hard root mean squared error can be calculated as:

$$RMSE^{hard}(T, \mathcal{P}_\phi) = \sqrt{\frac{\sum_{t=1}^{T} \sum_{j\in S_t\cup\{0\}} (I\{j_t == j\} - \mathcal{P}_\phi(j|S_t))^2}{\sum_{t=1}^{T} (|S_t|+1)}}$$

Jointly with the RMSE metric the average relative position of a particular model after ordering all tested models (under the RMSE criteria) from best to worst is reported, starting at number 1. The metric is computed as follows: for each out-of-sample data instance, all the tested models are ranked according to their RMSE and then, for each model, the average of its position across all data instances is taken.

Finally, the results based on the mean absolute percentage error (MAPE) are also reported. Since these results are generally similar (in relative terms) to the RMSE results, the presentation of those is relegated to Appendix 3.

3. PACKAGE IMPLEMENTATION

The last section introduced several discrete choice models and estimation algorithms. Despite the substantial amount of literature for these methods, most current state-of-the-art procedures lack publicly standard packages for academic or commercial purposes. A fact that might explain this phenomenon is that most of the assessed methods were developed in the last few years (see Section 2.2). Consequently, a custom package was crafted to execute the numerical experiments that follow the rest of the work. This chapter aims to explain some of the considerations taken into account during the implementation of these algorithms.

Firstly, demand estimation is an active area of research, and this field is no exception for machine learning rising popularity (see, e.g., Zhao et al. [70]). As a result, new models and variants are frequently proposed. Flexibility for adding new procedures in the future and during the work was the first desired feature. Secondly, in large-scale projects storing raw results for each model is greatly valued to compute ex-post statistics without rerunning the experiments. This allows researchers to decouple the estimation of the models from the quantitative analysis of the results.

The code was designed to have these two features into account: (i) introducing new models with ease and (ii) allowing raw experiments output storage (e.g.: info about computational time, optimized parameters vector for each model).

3.1 Chosen language

The chosen language was Python. This language has become one of the most popular choices for data science in the last few years. This widespread adoption means that more online packages were available to leverage implementation and that more researchers will be able to collaborate and replicate the results in the future. Python is an object-oriented programming language; this implies that it supports design patterns that can be applied to achieve greater flexibility. Moreover, the application uses hierarchies to avoid repeated code or behavior, making it easier for developers to add new models.

3.2 Code architecture

The code architecture in OOP programs is a reflection of the domain they try to describe. The concepts and relationships that best describe the real-world problem that is being solved are poured into the system through abstract structures called objects and classes

so that developers can relate to the real-world notions in the code. This seemingly obvious approach makes the application simpler to understand and modify by others. The popular UML diagrams [71] are commonly used to describe the system structure and share it with other programmers. Next follows a set of UML class diagrams that describe how the main discrete choice concepts were reflected in the code. These concepts include: transactions, models, estimation methods, and optimization problems. The discussion over the code architecture will also be helpful to explain how the intended goals for the package were achieved. For example, class hierarchies were used to avoid repeated code and describe shared behaviour between concepts. These hierarchies are also referred to as common interfaces, and allow each of the models (Markov Chain, Latent Class, Ranked List) to share functionality (RMSE computation, Axiomatic validation). These properties in the architecture will be pointed out throughout the next sections.

3.2.1 Transactions

The concept of the transaction is the simplest. It represents the outcome of a consumer faced with a purchase decision. When a consumer is shown an assortment S, it decides to purchase a product j (the no-purchase is just a particular case of a product). The tuple (j, S) is the reification for the transaction. Instances of this class store the data needed to fit the model's parameters during the demand estimation process.

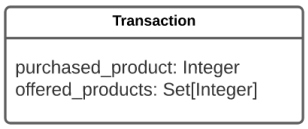


Fig. 3.1: A transaction is a purchased product and a set of offered products.

3.2.2 Models

A model's primary responsibility is to give structure to its probability function (see Section 2.1). That is, the model needs to compute $P(j|S)$ for each transaction (j, S), representing the probability of purchasing j when the set of products S is observed. The output of this function is manipulated through a parameters vector θ that is fine-tuned via estimation procedures to improve its predictive performance. The dependence on this parameters is usually shown explicitly through the notation $P(j|S, \theta)$. This behaviour is reflected in the class Model through a set of methods for computing probability, and accessors for the model's vector parameters.

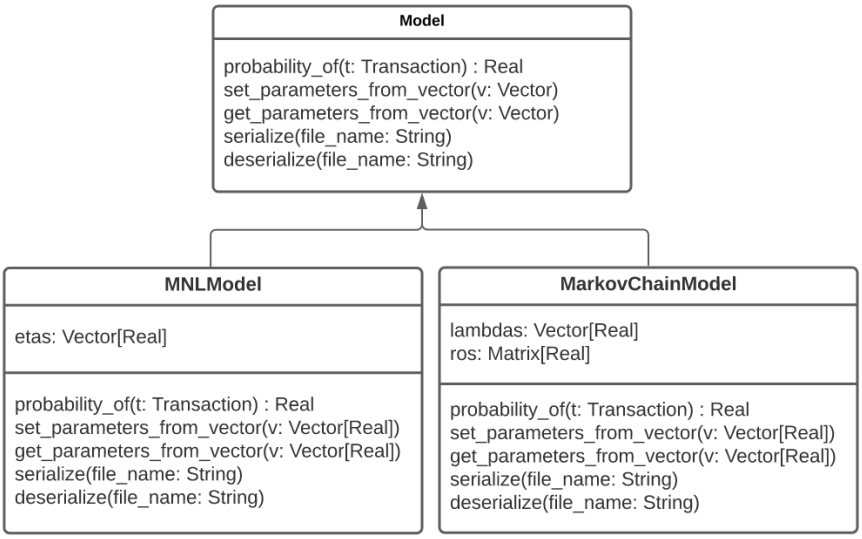


Fig. 3.2: Models share a common interface. They have methods for computing the purchase probability function, interact with their internal parameters and serializing. However, they have their own attributes (e.g.: λ and ρ for Markov chain.)

Notice that in figure 3.2 the MNLModel and MarkovChainModel share a common interface Model. This allows flexibility in adding new models while avoiding repeated code in other places. For example a maximum likelihood estimator can be used with any model as long as it provides this standard interface. There's no need to implement new estimation procedures each time a new model is added. This design also allows general tests to validate the axioms for any model or the computation of multiple metrics.

It must be noted that each model can have different internal representations for their parameters: vectors, matrix, or other data structures. For example the Markov Chain model uses a matrix to store its transition probabilities. However, models have to provide a procedure to transform these onto a single one-dimensional vector, mainly for its manipulation during optimization.

Finally, models are required to define serializing methods in order to store their internal state in files. This process is useful for decoupling metric computation from actual estimation. For instance, an optimizer can fit the model's parameters vector to a transaction set; this vector and its model can be serialized and stored in a file. Then, at any moment, this model can be loaded from the stored file, and additional metrics or further optimization on the parameters can be computed.

3.2.3 Estimation methods

An estimation method is an algorithm that modifies the parameters vector of a model to improve its predictive performance. As would be expected, they need a model and a set of training transactions as input. In turn, its output is a model with trained parameters. There are well-known estimation methods such as maximum-likelihood estimation or least-squares. These general procedures can be applied straight to all of the models, however, it is worth noting that some state-of-the-art methods have particular implementation details that have to be specifically handled. For example, the ranked list model has $O(n!)$ parameters and in practice a column generation method is required (see van Ryzin and Vulcano [60]). Although this method has particularities, it shares a common behavior with most of the estimation methods. They are iterative procedures that improve step-to-step the model's parameters vector.

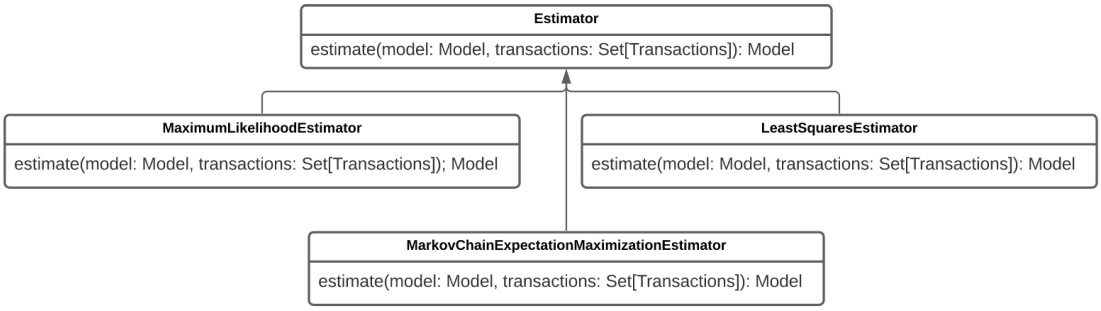


Fig. 3.3: Estimation methods share a common interface to avoid redundant code.

For this reason a class hierarchy was included in the code structure. In figure 3.3 we can see that the MaximumLikelihoodEstimator and LeastSquaresEstimator classes both inherit from the Estimator class. This means that useful behaviour such as tracking and saving the procedure progress at each iteration is automatically added to each new estimator. This advantage also applies to expectation maximization or market discovery algorithms.

3.2.4 Optimization problems

Most of the estimation methods solve one or multiple optimization problems. Maximum likelihood estimation involves maximizing the log-likelihood of a model for a set of transactions. The ranked list market discovery algorithm solves a mixed-integer linear program to choose new preference lists. These problems share behaviour that can be reflected in common interfaces. Also, optimization problems are solved with one of several standard

packages. The chosen package often varies between projects because of attributes such as quality, cost, or computational time. Each of these libraries has its particular interface, making it difficult to translate a problem from one package to another. To account for this, abstraction layers were included, allowing multiple optimizer backends to work with the package. For example, during the research ipopt [72] and scipy [73] were tested and implemented. On a later stage, ipopt was discarded because it did not provide functionality for computing the objective function's gradient numerically, however the description of the optimization problems didn't have to change.

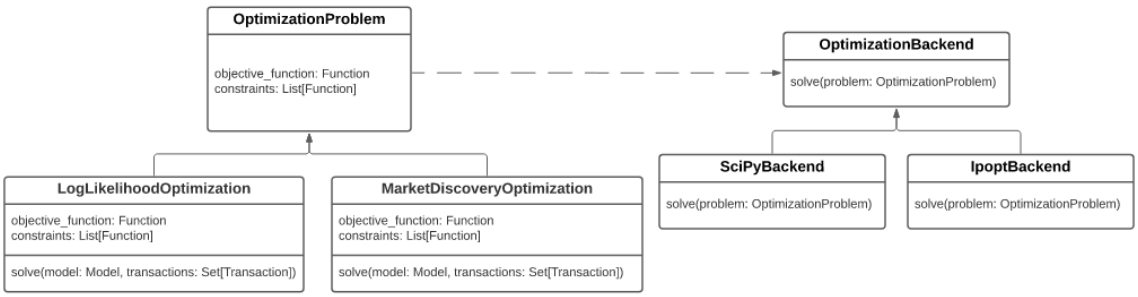


Fig. 3.4: An abstraction layer allows the usage of multiple optimization backends.

The main attributes of an optimization problem are the objective function and constraints. These manipulate the problem variables to attain a specific goal (maximization or minimization). Once these general requirements are coded, an optimization backend translates them to work with different packages depending on the programmers needs. Then, the third-party package actually does the optimization, and the solution is finally translated once again to the main program.

3.2.5 Classes diagram and relationships

To better understand the relationship of the previously mentioned classes, here is a complete UML class diagram. The striped line means dependence. For example, Transaction existence does not depend on any other class. However, Model depends on Transaction to exist: its purchase probability depends on a Transaction. Following the same reasoning, Estimator needs Transaction and Model to operate. Finally, optimization problems can depend on different objects depending on their structure and purpose.

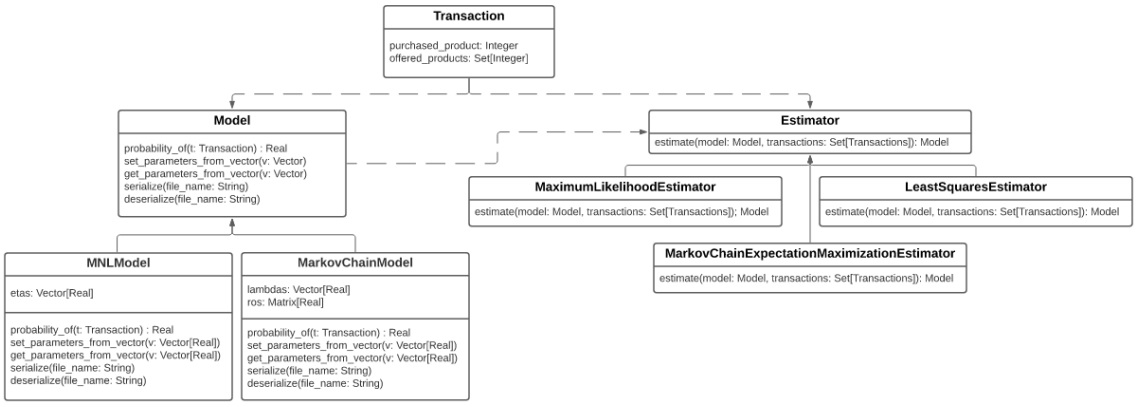


Fig. 3.5: Full class diagram.

As stated previously, the package's main goals were: (i) to achieve flexibility in the addition of new models and (ii) to have the ability to store the results allowing ex-post computation of statistics. Both of these were tackled by the usage of common object-oriented design patterns. Common interfaces were introduced to handle functionality such as metric computation, estimation, and optimization for all of the models. This means that in order to assess a new model, only its purchase probability and a parameter's vector are needed. This flexibility, hopefully, will be useful for future research in Machine Learning based methods. Finally, the model serialization provides functionality to store its parameter vector. This made the research process simpler by allowing to pause and resume optimization of models and decouple the estimation from the result's analysis.

3.3 Online availability

Ultimately, it is noted that this code is publicly available at GitHub [1] and was used for the numerical results shown in the subsequent sections and appendix. Also, at the time of this writing, there is active research based on this code by other researchers.

[1] Code available at `https://github.com/ajgara/choice-models`.

4. HIGH-PERFORMANCE COMPUTING

4.1 Computational load

This work assessed nine different models using instances varying: the amount of data, consistency of preferences, market share, price structure, assortment size, ground truth, and estimation methods. Some of these estimation procedures implied intensive computational load, as can be seen in chapter 5. Each combination of model and estimation procedure had to process millions of transactions from multiple instances. An average computer could not process this amount of data in an entire year. Even if one could afford to wait one year for this set of experiments to finish, the research took part as an iterative process of observing and modifying the experimental set-up via trial and error. This inconvenience meant that the whole process could have taken several years; an efficient alternative had to be implemented.

4.2 High-performance computing

This problem is common in several fields such as Machine learning or Big Data, where terabytes of data have to be processed in a small gap of time. Several commercial and academic tools were developed in the last few years to tackle this problem. Cloud computing services such as Google Cloud or Amazon AWS provide on-demand computational capacity to execute these intensive tasks. These are typically called high-performance computing centers.

A high-performance computing environment (HPC) is a large set of computer systems combined through a straightforward interface that provides the capacity to run a large number of tasks simultaneously. Tasks that are impossible to run on a personal computer can sometimes be adapted to execute on HPC.

In this case, most of the execution time could be easily parallelized, as models, methods, and instances were independent one from another. The same instance could test two methods at the same time. Thus, an HPC could be used to reduce computational time significantly.

4.3 Spartan HPC

Spartan [74] is an HPC center at the University of Melbourne. It consists of one central node and more than a hundred computing nodes connected through high-speed network-

ing. The University of Melbourne generously provided access to their infrastructure for this research project.

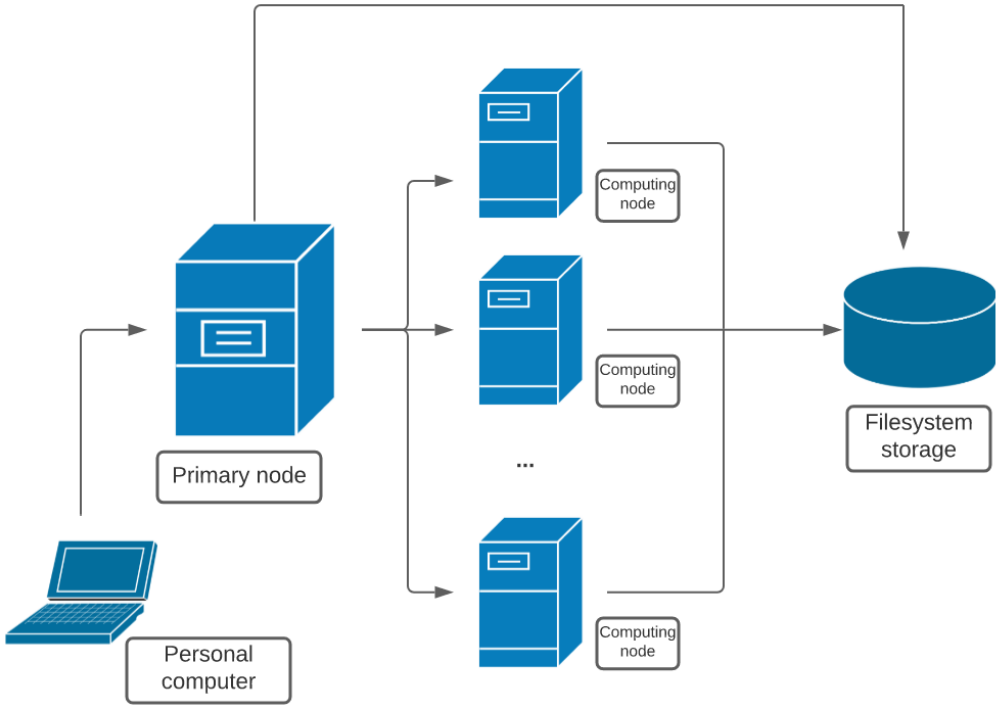


Fig. 4.1: The primary node can be accessed through SSH. The computing nodes execute the jobs submitted by the primary node and store the results in a shared filesystem.

A standard SSH connection grants access to the system's central node—there, a console allows to install the required packages and download the datasets in a shared filesystem. This primary node uses SLURM, an open-source workload management system that also presents functionality for job scheduling. This tool provides the necessary interface for running experiments on the rest of the cluster's nodes.

Before running an experiment, a task (or job) has to be defined. A task is a plain text file specifying a set of resources, a command, and expected usage. This text file follows a specific language determined by SLURM. For example, one can ask for a node with 4GB of RAM to execute a specific command for 3 hours. After its description, the task is submitted from the central node to a priority queue where it awaits a free node. Once a node is available, it executes the task and writes the results to disk storage (also administered by the University of Melbourne).

```
[agustin@spartan-login3 ~]$ squeue --user=radewale | head -n 15
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
          19379301      long 100_500I radewale  R 11-05:21:57      1 spartan-bm027
          19379249      long 10_500IR radewale  R 14-11:54:38      1 spartan-bm027
          19379252     snowy 10_500IR radewale  R 17-10:34:38      1 spartan-snowy008
          19379253     snowy 10_500IR radewale  R 17-10:34:38      1 spartan-snowy008
          19379275     snowy 10_500IR radewale  R 17-10:34:38      1 spartan-snowy009
          19379297     snowy 100_500I radewale  R 17-10:34:38      1 spartan-snowy013
          19370291     snowy 100_700I radewale  R 18-17:21:39      1 spartan-snowy007
          19303613      long 100_900I radewale  R 21-19:53:27      1 spartan-bm027
          19379060     snowy 10_600IR radewale  R 18-17:21:39      1 spartan-snowy010
          19379085     snowy 100_600I radewale  R 18-17:21:39      1 spartan-snowy012
          19379087     snowy 100_600I radewale  R 18-17:21:39      1 spartan-snowy012
          19303589      long 10_900IR radewale  R 21-22:18:31      1 spartan-bm027
          19379168      long 1_500IR_ radewale  R 15-18:22:11      1 spartan-bm027
          19379220      long_0.01_500 radewale  R 15-18:22:11      1 spartan-bm027
```

Fig. 4.2: List of jobs in spartan queue, including username and execution time.

SLURM also provides a set of instructions to query and modify the state of active jobs. The scheduler gives each task an ID; this allows performing operations over the specified job. One useful operation is cancellation. If a mistake is found in the running code, it is possible to interrupt the job execution. This service is useful for reducing computational waste in the priority queue. Another useful operation is the querying of the job's state. Tasks have different states: awaiting execution, executing, canceled, or finished. It is possible to check if the task has already started or finished. SLURM can be set-up to provide notifications via e-mail to avoid polling over the job's state.

Although SLURM is simple, it lacks the proper mechanisms to deploy thousands of experiments for execution. An encoding for the instances and a Python utility was designed to avoid this manual task.

4.4 Instance encoding

The package introduced in the last chapter was installed on Spartan. Also, the described datasets in chapter 5 were generated with a local computer and uploaded. Finally, the only step remaining was creating a task to run an experiment for each combination of model, estimation method, and data instance.

Although all instances could be executed independently one from another, they were grouped according to properties such as ground truth and number of periods to simplify the debugging and execution process. [1] These resulted in more than a hundred different experiments. Each of these experiments was given a descriptive human-readable name and an ID number to make the process easier.

[1] The next chapter provides details over the variables that were considered in the experiments (e.g.: ground truth, price structure, number of periods)

Finally, all of the instances belonging to a specific experiment were placed on the same folder. This encoding made it possible to reference, in a human-friendly way, any of the thousands of instances available.

4.5 Python utility

As mentioned, a plain text file has to be created for task submission on SLURM. A utility was developed to reduce the effort of creating by hand these files.

```
#!/bin/bash
#SBATCH -o 31_10x1_structure_100_lists_600_periods_50_share.mnl_max.out
#SBATCH -e 31_10x1_structure_100_lists_600_periods_50_share.mnl_max.err
#SBATCH -p cloud
#SBATCH --time=15:00:00
#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --mem=2048
#SBATCH --cpus-per-task=1
#SBATCH --mail-user=ajgarassino@gmail.com
#SBATCH --mail-type=ALL
module load Python/2.7.12-goolf-2015a
module load CPLEX
export PYTHONPATH="/home/agustin/lib/python/"
python estimate.py mnl_max 31_10x1_structure_100_lists_600_periods_50_share/data scipy
```

Fig. 4.3: An example of a task submission file: CPU, RAM and time requirements are provided for SLURM to schedule the job using the priority queue. An environment with pre-installed python packages is also loaded.

This utility could create task submissions for a set of experiments, estimation methods, and models. Specifying only IDs of experiments and estimation methods all of the necessary SLURM files would be generated. After that, a batch of tasks could be submitted to the priority queue and monitored through the standard commands.

After being executed, the experiment's task stored the serialized results for each model on a folder based on their encoding. These results included computational time, objective function at each iteration, and standard metrics in the literature such as RMSE or MAPE. They also included the raw estimated parameters for each model, allowing the computation of ex-post statistics.

Throughout the project, several performance metrics were tested (e.g.: hit-rate or AIC). For this reason, the utility included a mechanism to compute these metrics in parallel after the nodes executed the experiments.

4.6 Reduced computational time

Overall, this work tested 8,700 instances, with 22 million transactions in total. Different models processed each of these instances, adding up to 11, of which 9 ended in the final

project [2]. Each of the models was estimated using both maximum-likelihood and least-squares. Additional methods were coded for the ranked list (EM), latent class (Frank Wolfe), and Markov chain (EM) models. A total of 25 procedures had to be executed for each of the 8,700 instances. Each estimation took an average time of 680 seconds. All this computational time adds up to 4 years and a half just for estimation (not including assortment optimization; e.g.: see sections 6.2.1 7.2).

Each algorithm has 30 minutes to complete its estimation (e.g.: see appendix 1). So in practice, if all the experiments could execute simultaneously, the final computing time would be 30 minutes. However, HPC centers have a restricted number of nodes, and many projects share the same resources. The result, however, is that all of the experiments ended running in a time-span of a few months.

[2] As stated in Section 2.2, two of them were simple benchmark models to attain more confidence in the results.

5. PERFORMANCE UNDER SYNTHETIC PREFERENCES

In order to assess the performance of the different discrete choice models using synthetic preferences, two large families of instances that differ in terms of their product structure were created. In the first family, called *random*, there is no ex-ante natural preference of each product over any other product. In the second family of instances, there is a natural preference order (i.e., vertical differentiation) between some of the products. In particular, it is assumed that the retailer may potentially alter the prices of its products, where each product can have three potential prices.

All these instances are constructed using the rank list-based (also called *stochastic preference*) model. That is, each instance consists of a preset number k of strict preference lists among all alternatives and a probability distribution over those k preferences. Details about the construction of the instances is given below. Of course one could have created different instances using other discrete choice models. Nevertheless, using such model has at least two benefits. First, it has been shown that the stochastic preference model is indeed equivalent to the general random utility model described in Section 2.1 (e.g., see Block and Marschak [43]), subsuming all the choice models benchmarked in this work. Second, the model has a realistic and simple interpretation: consumers can be clustered into k unobservable types, and each type has a strict preference among the alternatives [1]. In the appendix, the results of additional experiments where the ground truth comes from the latent class MNL (Section A4) and from a nested logit with prices (Section A5) is reported.

5.1 Description of instances

This section starts with unstructured *random instances*, where any product could potentially be selected over any other product, followed by the *structured instances* where there is natural order between products offered at different price points.

5.1.1 Random instances

The first stage in the data generation process is the creation of a ground truth. There are $n = 10$ products. Each data instance has $k = 10$ or $k = 100$ preference lists. In order

[1] In fact, the model does not restrict consumers to belong to a single consumer type. It can also be interpreted in the context of individuals having different probabilities of selecting a type (i.e. a strict preference) and randomly choosing one in each store visit.

to force some consumers to select the no-purchase option in every assortment, the first preference list (number 1) contains product zero (i.e., the no-purchase option) as the first alternative. The probability assigned to this list is $p_1 \in \{0.8, 0.5, 0.1\}$, depending on the instance. Each of the remaining preference lists $(2, \ldots, k)$ is selected uniformly at random out of all the possible permutations of the 11 alternatives, only excluding those whose first choice is the no-purchase option. In addition, a random number u_i between 0 and 1 was generated for each $i = 2, \ldots, k$, and then assigned $p_i = \frac{u_i}{\sum_{j=2}^{k} u_j + p_1}$ to the probability of choosing the preference list $i = 2, \ldots, k$.

The reason to construct ground truths with $k = 10$ and $k = 100$ preference lists is to understand how the different models perform depending on the complexity of the consumers preferences. For example, does the relative performance of a simple model such as the MNL improves or deteriorates when $k = 100$ in comparison to when $k = 10$? On one hand, a model with $k = 100$ preference lists is more complex than one with just $k = 10$ lists, which might indicate that its performance may deteriorate. At the same time, when the number of different consumer types is large, one can think as if the segments are less clearly divided and the preferences are more homogeneous, so that the model may be well approximated by a smooth function, which can be beneficial for simple models such as the MNL. On the contrary, when the number of different consumer types is small, one could think that the segments of the population are clearly divided, leading to non-smooth preferences and hence more challenging to approximate by a simple model.

The second stage in the data generation process, once a ground truth is constructed, is to generate in-sample transaction data. The data consists of T periods, with $T \in \{30, 75, 150, 300, 600\}$, each one containing 10 transactions. Each period is associated with a randomly generated assortment or offer set $S \subset \{1, \ldots, 10\}$ such that $3 \leq |S| \leq 6$. Each of the 10 transactions for a specific period is obtained by randomly selecting one of the k preference lists (using the known probability p_i of each list) and then choosing the first alternative appearing in the preference list that belongs to $S \cup \{0\}$, where S is the offer set associated to that period.

Using the in-sample generation procedure, 60 instances for each combination of k (10 or 100), T (30, 75, 150, 300, or 600) and probabilities for the no-purchase list p_1 (0.8, 0.5, or 0.1) were generated. Observe that from the no-purchase probability p_1, one can establish that the expected fraction of purchases is at most $(1 - p_1)$. This fraction is called the *market-share* (under full availability), and this is how the instances are distinguished in the result tables: 20% ($p_1 = 0.8$), 50% ($p_1 = 0.5$) or 90% of market share ($p_1 = 0.1$). Also, the instances are generated in a nested way with respect to the value of T. That is, the instances for $T = 75$ are a superset of the instances with $T = 30$, after adding 45 extra

periods. This incremental construction guarantees that there is value-added information in the bigger instances.

The relative small size of the offer sets in the in-sample instances is justified by the purpose of enforcing customer substitution behavior as a means to evaluate the ability of the different models to capture it. Later, in Section 5.2.4, the impact of a wider product offer is analyzed by enlarging the available assortments.

Regarding the performance assessment of the different models, as explained in Section 2.5, the soft RMSE takes into account all possible assortments and therefore there is no need to generate out-of-sample transactions to assess performance. However, in order to reduce over-fitting, all the 600 assortments that may appear in the in-sample were removed from the out-of-sample. The same approach was replicated in all the experiments that follow in this section and in the ones in Section 6.

5.1.2 Structured instances

A relevant feature of the previous *random instances* is that all products can in principle be selected with positive probability. This is because the preference lists were generated uniformly at random so no product was given an ex-ante higher preference.

In practice, however, managers face the problem of predicting consumer choices over assortments whose products have considerable different levels of attractions for consumers. In order to account for this, a setting in which the firm has $N = \{1, \dots, m\}$ different product types that could offer to consumers is considered. Furthermore, the firm also assigns a selling price to each product at every point in time. Here it is supposed that for every product i, the selling price belongs to the set $R(i)$ where $R(i) = \{r_{i,1}, r_{i,2}, r_{i,3}\}$, with $r_{i,1} \leq r_{i,2} \leq r_{i,3}$. In words, each product can be offered at one of three different prices associated to it. This structure of vertical differentiation is standard in the airline industry, where an economy seat for an itinerary can be offered at several prices or fares. Although different fare classes may have different attributes, in many cases the only difference between two fare classes is the price at which they are sold.

One may interpret this setting as if the universe of potential products $\mathcal{N}$ has size $n = m \times 3$. That is, each alternative is a pair (i, r) such that the original product type is $i \in N$ and its price is set to $r \in P(i)$ (thus, the same item with different prices are treated as different items). It is further assumed that if the original product type $i \in N$ is offered, it is offered at a unique price. Finally, it is also assumed that consumers always prefer a lower price to a higher price for any given product.

At this work it is of interest to study the purchasing probabilities of the different products given that a subset of them is offered at no more than one of the different

possible prices. Observe that under this setting the offer set contains at most m products.

The ground truth generation proceeds as follows. A set of five products ($m = 5$) is considered. As in the case of the random instances, stochastic preference instances with either $k = 10$ or $k = 100$ preference lists are generated. The first list is also fixed to have the no-purchase option as the first choice and vary the probability associated to that list between 0.1, 0.5 and 0.8. Since each product has three different prices, each preference list has length 16 (note however that the way in which the alternatives are ordered in a preference list after the no-purchase position is irrelevant).

Once a list is randomly generated, it enters a process here called *rationalization*, whose purpose is to enforce "low over high" price preferences for a given product type. The procedure goes over a preference list from top to bottom. If there is an alternative (i, r_k) with $k = 2, 3$ and any cheaper version of product i has not appeared in the list so far, it inserts that (or those) alternative(s) immediately before the alternative (i, r_k). In this way the items are always preferred at cheaper prices.[2]

In order to generate the in-sample transaction data for a given stochastic preference, a similar process to the one described for the random instances is followed. That is, there are T periods (with $T \in \{30, 75, 150, 300, 600\}$), each containing 10 transactions. The offer set S associated to each period is randomly generated such that $3 \leq |S| \leq 5$. Naturally, it is also imposed that the offer set cannot include the same product under two different prices.

5.2 Results

Overall, the different discrete choice models were compared over 3,600 instances. These instances were constructed by varying different parameters of the data generation process such as the volume of historical data, the market-share, the number of preference lists, and whether products have a price structure or not. This highly intensive computational task, which amounts to over a year of CPU time, was executed in parallel in a cluster at the University of Melbourne.

In order to facilitate the interpretation of the results, an aggregate summary of the performance of the different models when varying different data features is shown. The results here are based on the soft RMSE performance measure and the average ranking position of each model across all the instances. The estimates were obtained via maximum likelihood according to the implementations described in Section 2.2. At the big picture level, the results under least squares are similar and these are relegated to Appendix A2.

[2] For instance, if for a given preference list product 2 is first offered at price r_{23}, then options $(2, r_{21})$ and $(2, r_{22})$ are inserted immediately above $(2, r_{23})$.

5.2.1 Impact of volume of historical data

This section compares the performance results obtained by the different models under two different scenarios. Small historical data ($T = 30, 75$) and large historical data ($T = 300, 600$) including both random and structured instances. Figure 5.1 shows the overall average performance of the soft RMSE metric for the different choice models. Each scenario (small and large historical data) contains a total of $1,440$ instances. The number displayed above each bar in Figure 5.1 is the average relative position of the corresponding model across all instances, as explained in Section 2.5.

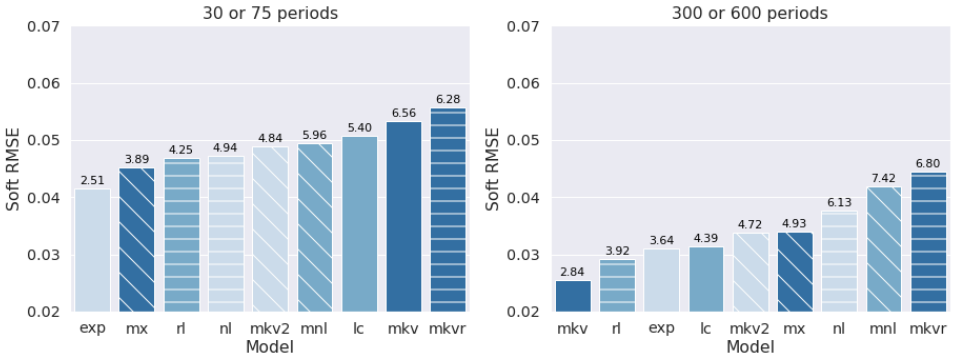


Fig. 5.1: Average soft RMSE across all instances with 30 and 75 periods (left), and, 300 and 600 periods (right). The average performance position of each model is displayed above the corresponding bar.

First, and naturally, the performance of *every* choice model improves considerably as the number of in-sample periods gets larger. For scenarios with small historical data, it is noted that the exponomial model has a clear performance advantage. The exponomial is then followed by the mixed MNL, the rank based model, and the nested logit (in that order, but with very similar performance). The three models that perform worse are the latent class model, the Markov chain, and the reduced Markov chain. All these models have more parameters, so it is expected that they would suffer from some degree of overfitting in scenarios with small historical data. Noticeably, even the LC-MNL performs worse than the MNL model despite being more general than the latter.

For the scenarios with large historical data ($T = 300, 600$), the relative performance changes drastically. In this setting, the models which perform best are the Markov chain and the ranked list (in that order), with quite significant improvements with respect to their small historical data counterparts. The positive change stems from the fact that these two models have many variables and therefore, they can better exploit the availability of more historical data to fit the parameters. In Figure 5.2 it is depicted how the Markov chain model (with $O(n^2)$ parameters) overpasses the exponomial (with $O(n)$ parameters)

as the number of in-sample periods increases.

Note that the reduced Markov chain, with the superimposed arbitrary pattern of substitutions, performs the worst independently of the data volume (though still improves as the data volume increases). Also, as a general observation, both the ranked list and the exponomial models perform among the top 3 irrespective of the data volume. This is expected for the ranked list model (recall that the ground truth was generated using this model), but somewhat surprising for the exponomial, the model revisited by Alptekinoğlu and Semple [25] but relatively understudied in the OM literature.

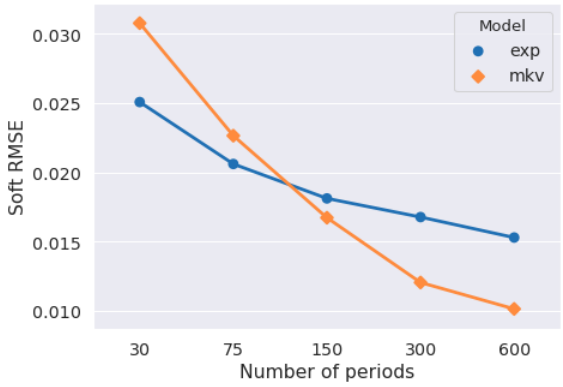


Fig. 5.2: Impact of volume of historical data. Average soft RMSE for the set of instances with 20% market share, random structure and 10 preference lists. It can be seen how the Markov overpasses the exponomial as the in-sample data volume increases.

5.2.2 Impact of structured preferences

Here, the performance results obtained by the tested models depending on the structure of the preferences is studied. In order to isolate the differences, only instances with large historical data ($T = 300, 600$) were considered. A priori, it is not clear how a model may perform under each type of instance. The reason is that the structure of the underlying consumer preferences under different data generation processes is not the same: Under the random structure (10×1), preference lists are less restricted than in the price structure (5×3) instances. Figure 5.3 shows the aggregate performance of the different choice models under both settings.

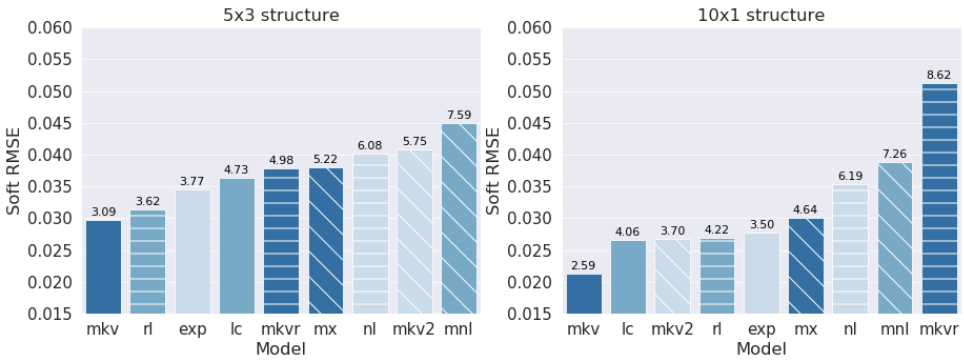


Fig. 5.3: Impact of structured preferences. Average soft RMSE over instances with 5×3 price structure (left), and a 10×1 price structure (right). The results are aggregated across in-sample instances of 300 and 600 periods.

As one can observe from Figure 5.3, switching from structured (i.e., 5×3) to random (i.e., 10×1) instances induce all models to reduce their average soft RMSE. The single exception here is the reduced Markov chain model which was specifically tailored to address the 5×3 structured preferences.

Here, consistently with the observation from Figure 5.1, the Markov model performs noticeably well. The very good performance of both ranked list and exponomial over both instance types can also be observed.

5.2.3 Impact of the consistency of the underlying preferences

The performance results obtained by the different models depending on the number of different consumer preferences (10 or 100 preference lists) is now compared. The 10-list case is referred as the *more consistent* scenario, since the probability of having customers of a given type is (on average) higher than the probability of having customers of a given type in the case of 100 lists. In order to isolate this effect, only instances with large historical data ($T = 300, 600$) are considered. The main finding here is that relative performance of the choice models strongly depends on the consistency of the consumer preferences. Figure 5.4 shows the aggregate performance of the different models under the two scenarios.

First, it is observed that the aggregate performance of every model increases when we move from 10 lists to 100 lists. A priori, this might seem counter-intuitive as consumer preferences under the 100 list instances may appear as less consistent compared to those with only 10 preference lists. However, instances with a lower consistency of consumer preferences are slightly less sparse (i.e., there are fewer zeroes in the distribution over

the $O(n!)$ potential preference lists[3]), which would help choice models to make fewer errors at the aggregate level. Recall that the preference lists in the ground truth are uniformly selected, in consequence the ground truth becomes more random as the number of lists grows. The argument is comparable to the one in the previous section: when switching from more to less structured instances, parametric models like the exponomial and the logit-based models show a substantial performance improvement.

Out of the best four models under each scenario (10 lists and 100 lists), the exponomial is the one that has the smallest number of parameters. In instances with 10 preference lists, the models that perform the best are the ranked list and the Markov chain followed by the LC-MNL and the exponomial. A clear advantage of the ranked list in these instances is that, given that the number of preference lists is rather small, it might be possible for it to replicate the choices of the ground truth model with a small number of preference lists.

Now, in instances with 100 preference lists something significantly different is observed. Perhaps quite surprisingly, the best model here is the exponomial, followed by the Markov chain and the mixed MNL. The relative performance of the ranked list model has deteriorated quite remarkably here. Naturally, it is much harder to recover the ground truth for the ranked list model as the number of underlying preference lists increases. On the other hand, the exponomial, which is a non-sparse model, has increased its relative performance considerably. To confirm this, further experiments were performed to check the relative performance of the different choice models on instances with a ground-truth of only two preference lists versus instances based on 100 preference lists (see Figure A9 in Appendix 3). Not surprisingly, it is observed that the ranked list model performs even better over 2 preference lists compared to 10 and 100 preference lists. On the other hand, the exponomial continues to worsen in terms of relative performance from 100 to 10 to 2 lists.

[3] Note that with $n = 10$, an instance with 100 preference lists implies that over 99.998% of all the $O(n!)$ possible lists have zero probability of being chosen. It is therefore the case that these instances are very spiky and far away from a non-sparse, non-spiky ground truth, although the case with 100 lists is *less consistent* than the case with 10 lists.

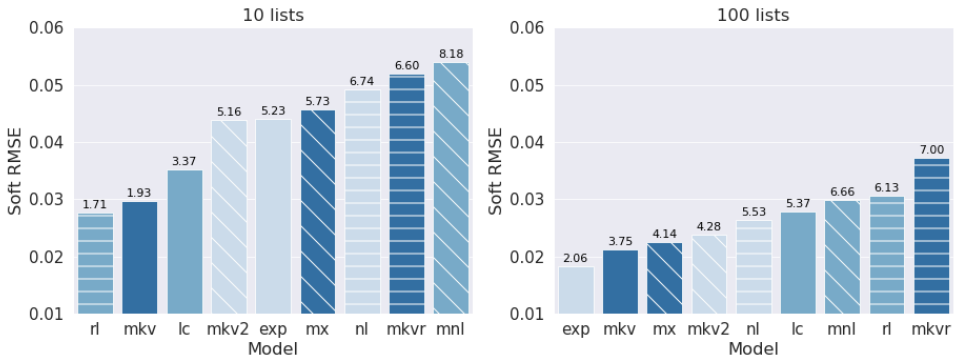


Fig. 5.4: Impact of the consistency of the underlying preferences. Average soft RMSE on instances with 10 preference lists (left), and 100 preference lists (right). The results are aggregated across in-sample instances of 300 and 600 periods.

5.2.4 Impact of the assortment size

The assortment size used so far has been between 3 and 6, which induces a high degree of substitution given that the full assortment size is 10. Here, the performance of the different choice models when confronted to bigger offer sets (of size between 7 and 10) is studied. Figure 5.5 illustrates this by only showing results over instances with large historical data ($T = 300$ and 600).

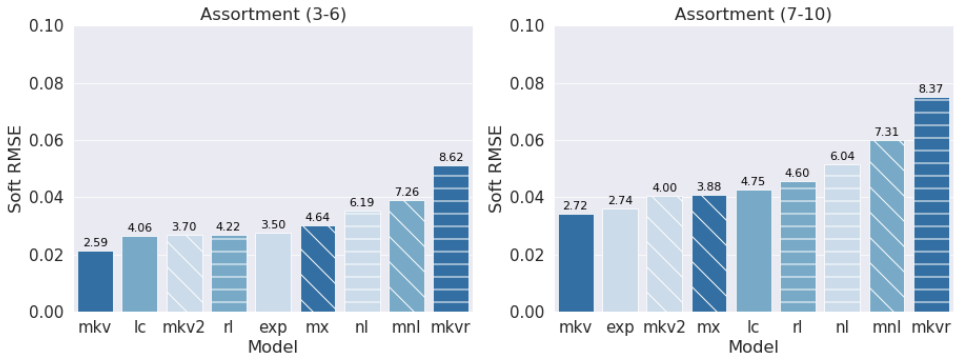


Fig. 5.5: Impact of the assortment size. Average soft RMSE on instances with assortments between 3 and 6 products (left), and between 7 and 10 products (right). The results are aggregated across in-sample instances of 300 and 600 periods.

When comparing the height of the bars in the left and the right panels, a first observation is that all models worsen their performance when enlarging the offer sets; that is, when the substitution behavior is less emphasized as customers tend to find their most preferred option. Under widely open offer sets the value of discrete choice models diminishes and customers display a behavior that becomes closer to the independent demand

paradigm.

Second, even though the relative order of the models' performance is not exactly the same, there is a cluster of six models which are the best performers in both panels. The Markov chain model leads the board throughout.

5.2.5 Computational time of the estimation process

In some industry applications (e.g., e-tailers, airlines), the estimation time plays a key role in the selection of a choice model. In Table 5.1 the time taken to estimate instances is averaged. In the first column (Small) all instances whose number of in-sample periods are either 30 or 75 is aggregated. For the second column (Large), all instances whose in-sample periods are either 300 or 600 is averaged. The total number of instances aggregated is 1,440 in each case. Due to implementation issues (specifically, not being able to stop an estimation procedure at a specific point), the time taken by an estimation method sometimes exceeds the time-out limit of 1,800 seconds. The percentage of the instances that reached the time-out is also reported.

One can see from the table that the MNL is by far the fastest to estimate. It is then followed by the nested logit and the exponomial, which take on average the same order-of-magnitude time and less than a minute even for the large instances ($T = 300$ and 600). The LC-MNL then follows with an average of 445 seconds for large instances. Finally, the slowest methods are the ones associated with mixed MNL, ranked list and Markov chain models (in that order). Recall that the mixed MNL was estimated using a computationally intensive method involving numerical simulation. The ranked list is estimated by solving multiple MIPs whose size increase with the number of periods in the in-sample. Observe that for the large instances, the estimation methods of the mixed logit and the ranked list reached the 30 minute time-out more than 98% of the time.

Complementary to Table 5.1, in Figure 5.6 the estimation time versus soft RMSE value for all the models under consideration is shown. Note that in each panel (i.e., small data with only 30 or 75 periods (left), and larger data with 300 or 600 periods (right)), we would like a model to be in the lower left corner. For a small number of periods, the exponomial model dominates the other ones. For a large number of periods, an efficient frontier defined by the Markov model, the exponomial and the MNL is identified. In fact, the exponomial again seems to provide a very good balance between these two performance measures.

Model	Small		Large	
	Avg Time	Time Outs	Avg Time	Time Outs
mnl	1 s	0.00%	4 s	0.00%
nl	6 s	0.00%	45 s	0.00%
exp	8 s	0.00%	48 s	0.00%
lc	76 s	0.00%	445 s	1.16%
mkvr	31 s	0.00%	893 s	37.10%
mkv	409 s	0.84%	1,018 s	49.64%
mkv2	574 s	11.12%	991 s	43.04%
rl	651 s	12.35%	2,189 s	98.55%
mx	1,669 s	79.80%	1,858 s	98.48%

Tab. 5.1: Average computational time taken to estimate all instances with an in-sample of 30 or 75 periods (Small), and in-sample of 300 or 600 periods (Large). In total, there are 1,440 small instances and 1,440 large instances. The Time Outs columns report the percentage of the instances that reached the maximum time-out limit of 30 minutes.

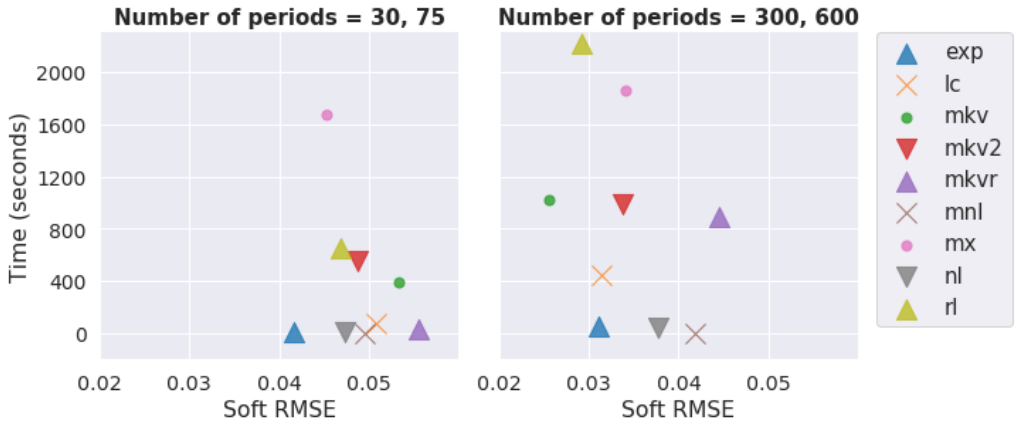


Fig. 5.6: Tradeoff between computational times and soft RMSE. Average across instances with 30 and 75 periods (left), and, with 300 and 600 periods (right).

To conclude this section on synthetic data experiments, it is highlighted that the approach based on preference lists studied here is quite general but at the same time lacks parametric structure. There are setups where one can arguably assume that customers choose in accordance with some structural framework, like a latent class MNL based on product features (e.g., a customer in one class may prefer a regular soda to a diet soda, different from a customer in another class). Even though this type of behavior is subsumed within the stochastic preference approach, two series of experiments were also executed to assess the models' performance under ground truths with a parametric structure. In the first parametric experiment, described in a Appendix A4, the ground truth is set to

be the latent class MNL. In the second parametric experiment (Appendix A5), a ground truth based on the nested logit model that explicitly accounts for prices in the consumers' utility function is set up. Results and related discussions are included therein.

6. PERFORMANCE UNDER REAL PREFERENCES

To add robustness to the comparison of the different discrete choice models, their out-of-sample performance is analyzed using two sets of real preferences. The first dataset contains bookings from five hotels, and the second one, preferences of individuals over 10 different sushi types. Both sets are publicly available and have been previously referenced in the empirical choice-related literature.

6.1 Hotel data

In this section the performance of the discrete choice models is evaluated on a publicly-available dataset containing booking transactions from five continental U.S. hotels (see Bodea et al. [75]). The booking transactions (mostly from business customers) have check-in dates between March 12, 2007, and April 15, 2007. The booking horizon for each check-in date recorded was of at least four weeks. Every time there was a booking, the rate and the room type availability information present was recorded for reservations made via the hotel or customer relationship officers (CROs), the hotels' web sites, and off-line travel agencies. Since the hotels are not in direct competition (they are not located in the same regional area), each of them is treated separately. A *product* is defined as room type (e.g., king non-smoking, queen smoking, suite type 1, etc). Following van Ryzin and Vulcano [60] and Şimşek and Topaloglu [42], some preprocessing was performed to the dataset. Specifically, the products that have less than 10 purchases were removed. For some transactions (i.e. bookings) in this dataset, the set of available products were not available. These bookings were not considered. In addition, transactions where the purchased product is not among those declared to be available were removed. For the remaining bookings (i.e. those in which the available products are known), the estimation procedures for the different choice models was executed. For each real transaction, four no-purchase transactions using the same offer set were incorporated. For each hotel, 80% of the transactions was randomly selected as in-sample. Table 6.1 provides the main relevant features of the hotel data used for this experiment. It is noted that the fare (or price) wasn't used. The transaction and availability data after preprocessing, and the defined customer types, is provided as a supplemental e-companion to this manuscript.

In Figure 6.1 the gap for the hard-RMSE in the out-of-sample is shown for the different choice models calibrated on the five different hotels. Consistently with the results on synthetic experiments, it is observed that the exponomial, the latent class MNL, the Markov

chain and the ranked-list models perform (in relative terms) very well. In particular, the Markov chain model is the best in four out of the five hotels. In the remaining hotel, which is ranked second in terms of small historical data, the winner is the exponomial model. On the other hand, the MNL, nested logit and the mixed MNL models have a considerably lower relative performance. In this case, the results for the reduced Markov chain model are not reported since given that product prices in the dataset vary over time, it is difficult to map this model to a static product structure.

Feature	Hotel #1	Hotel #2	Hotel #3	Hotel #4	Hotel #5
Number of products after preprocessing, n	10	10	8	4	6
In sample transactions	5290	1845	5070	1100	1000
Out of sample transactions	1325	465	1270	275	255

Tab. 6.1: Basic information about the hotel dataset

Fig. 6.1: Gap in out-of-sample hard RMSE with respect to the best model for the hotel data. Each value is colored so that the lighter the color the better the choice model performed.

6.2 Sushi dataset

Here the performance of the discrete choice models is assessed using a dataset containing sushi preferences from 5,000 individuals (see Kamishima [76]). These preferences were obtained using a commercial web survey service as follows. First, each individual was shown on the screen $n = 10$ popular types of sushi: shrimp, sea eel, tuna, squid, sea urchin, salmon roe, egg, fatty tuna, tuna roll and cucumber roll (which are labeled with numbers from 1 to 10). Second, the individuals were asked to rank the sushi types according to their preferences (thus, each user provided one of the 10! permutations). It was not possible for the participants of this survey to rank the no-sushi option.

These preferences were used as basis to generate semi-synthetic datasets. Let $T \in \{30, 75, 150, 300, 600\}$ be the total number of periods in the in-sample. For each period t,

a random subset (assortment) S_t, with $S_t \subseteq \{1, 2, \ldots, 10\}$ and $3 \leq |S_t| \leq 6$ is generated. Next, for each of the in-sample periods, transactions are produced as follows: with 50% chance the no-sushi list is selected (i.e. a list whose most preferred option is the no-purchase), and with the remaining 50% chance, one list from the dataset is selected uniformly at random. Then the most preferred sushi out of those in S_t under the selected preference list is chosen as the purchased one. This is repeated 10 times for each period, so to have 10 transactions per period. These set of instances are called *top 10*, as each individual would purchase a sushi if the assortment contains any of her top 10 sushis.

In addition to this data generation, another set of instances that called *top 3* is also constructed, by repeating the same procedure except that now each of the 5,000 preference lists are trimmed after the third choice: it is considered that an individual would only buy a sushi if one of her top 3 preferred sushi types is in the assortment. Note that cutting the preferences after a small ordinal number in the ranking (in this case, 3) is an interesting case. The trade-off here is to allow for a diversity of consumer preferences (e.g. if we cut after the first choice there will be only 10 customer types) and at the same time account for the dependency between the propensity to purchase and the size of the assortment (if we cut after the ninth product, it is very unlikely that a consumer will not purchase a product when is being offered an assortment).

6.2.1 Estimation results

Figure 6.2 shows the average, out-of-sample, soft RMSE performance of the different models as a function of the in-sample size.

For instances constructed using the top-3 products and a large in-sample, it is observed that the best models are the Markov, followed by the mixed MNL and the exponomial. When the number of in-sample transactions is small, the best model is the exponomial, which is consistent with the synthetic experiments. In these instances it is followed by the MNL and the nested logit models. Although all models improve their performance as the volume of historical data increases, it is worth mentioning the impressive improvement of the Markov chain model, starting from a RMSE of 0.047 (30 periods) and then reaching to 0.015 (600 periods).

The results (in relative terms) on the experiments taking into account the top 10 products (i.e. without chopping the preference lists) are similar. The exponomial and the mixed MNL do very well when there are 30 periods while the best model when there are 600 periods is the Markov chain. In this case, it is the best by a large margin.

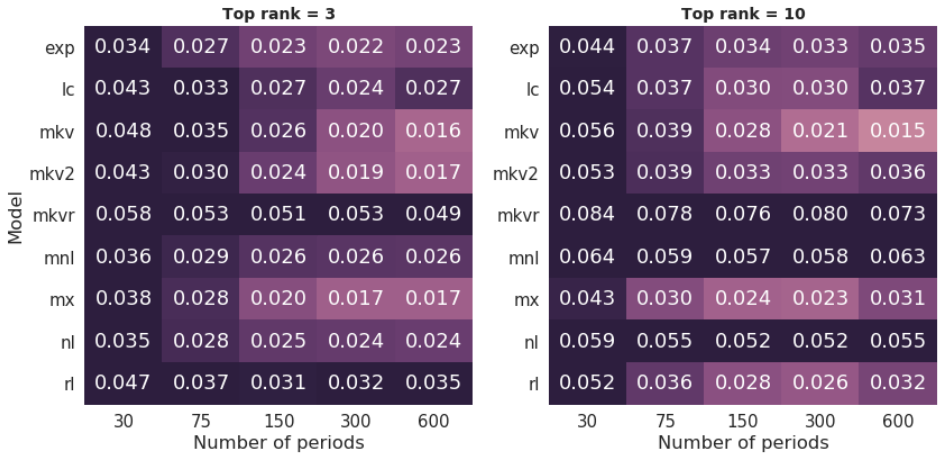


Fig. 6.2: Out of sample soft RMSE for the sushi example. Number of periods in the in-sample (horizontal axis) and choice model (vertical axis). Each value is colored so that the lighter the color, the better the choice model performed.

The impact on revenue

In retail operations and RM, one of the most important applications of the estimated choice models is to serve as input to conduct the critical assortment optimization task. Given revenues $r(j)$ for each product $j \in \mathcal{N}$ and ground truth probabilities, $\mathcal{P}(j|S)$, the assortment optimization problem to maximize expected revenues can be formulated as follows:

$$\max_{S \subseteq \mathcal{N}} \sum_{j \in S} \mathcal{P}(j|S) \cdot r(j). \tag{6.1}$$

Of course, the ground truth is not known by the decision maker but instead only sales transaction data coming from such ground truth is available. Consider a choice model $\mathcal{M}$ to be estimated from the available data, with corresponding estimated probabilities $\hat{\mathcal{P}}_{\mathcal{M}}(j|S)$.The assortment problem solved by the decision maker is

$$\max_{S \subseteq \mathcal{N}} \sum_{j \in S} \hat{\mathcal{P}}_{\mathcal{M}}(j|S) \cdot r(j). \tag{6.2}$$

Naturally, if $\mathcal{M}$ cannot properly describe the real consumer preferences, the sample-based optimal assortment $S^*_{\mathcal{M}}$ obtained from (6.2) may significantly differ from the ground truth-based optimal assortment S^* from (6.1). The purpose of this section is to study what the impact on the expected revenue is when the seller offers the assortment returned by (6.2) for the different tested choice models.[1]

[1] Besbes et al. [77] propose an alternative methodology to test the quality of an operational decision (e.g., pricing) based on a hypothesis test that they label "performance-based". This test checks the quality of an operational decision on a key driver (e.g., revenues) with respect to the best achievable scenario.

The sushi dataset described in Kamishima [76] also contains normalized selling prices of each sushi type. In this work those prices are taken as the revenue $r(j)$ for all j.[2] Because the sushi dataset does not contain the no-purchase option, it would always be optimal to offer only the most expensive product to maximize revenue. To avoid this trivial case, the consumer preferences were extended by adding the no-purchase as the $(k+1)^{th}$ most preferred option for each individual. In order words, if the assortment offered to a consumer does not contain any of her top k (with $k = 3$ or $k = 5$) options, then she would buy nothing.

In order to assess the model selection impact on revenue, 60 instances were generated using the sushi preferences, but truncated as just explained. For each instance, scenarios of small training data (with 300 periods) and large training data (with 6,000 periods) were generated with offer sets and transactions to train the different choice models, and then solve to optimality the corresponding assortment problem (6.2), leading to an optimal offer set $S^*_{\mathcal{M}}$ for each choice model. In Figure 6.3 the fraction of the optimal achievable revenues is contrasted with the revenue prediction error for each benchmarked model $\mathcal{M}$.

[2] It should be acknowledged that theses prices were not taken into account by the participants who ranked the sushi types. Thus, the optimal assortments obtained can be different to the best assortment in practice.

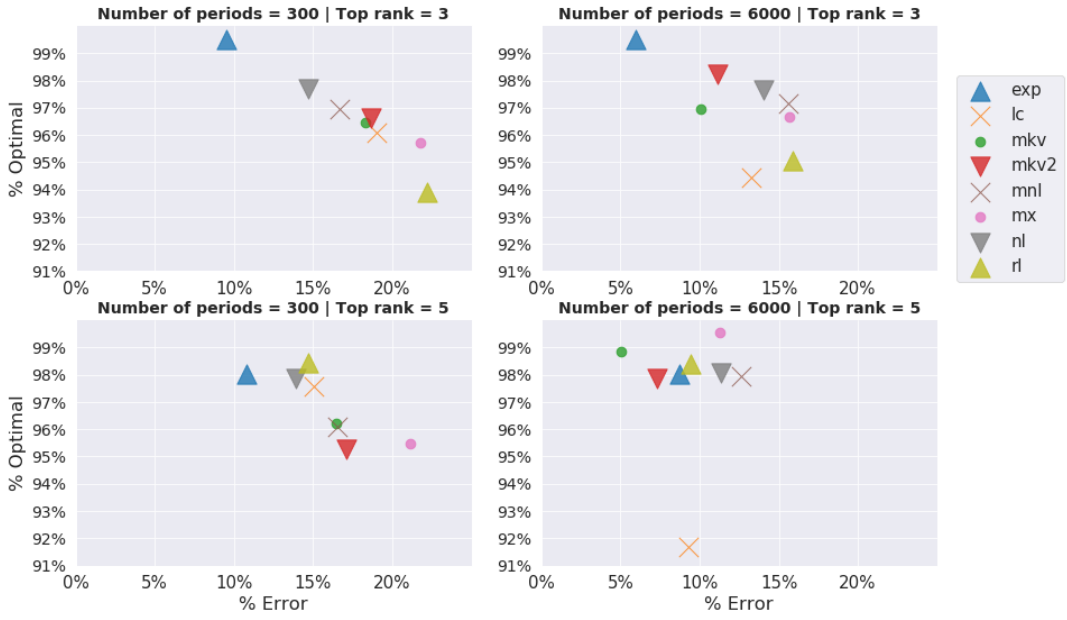


Fig. 6.3: Revenue assessment from different choice models on the sushi dataset. The horizontal axis accounts for the average specification error made by the modeler: It shows the percentage error between the predicted revenues by a model $\mathcal{M}$ from an offer set $S^*_{\mathcal{M}}$ and the expected revenues from $S^*_{\mathcal{M}}$ under the ground truth. The vertical axis accounts for the average percentage of the optimal achievable expected revenues under the ground truth, when both the true S^* from (6.1) and the computed $S^*_{\mathcal{M}}$ from (6.2) face the ground truth.

Ideally, a choice model would be placed in the upper left corner of each panel, implying that the revenue prediction error is negligible and the associated optimal offer set achieves the maximum expected revenue from the ground truth. The percentage error of the different models (horizontal axis) reflects an overestimation of the obtainable revenues from their proposed assortments, and this is particularly emphasized for the ranked list-based, MNL and mixed logit. It is noted that this is due to a selection bias, since for each model the assortment with highest ex-ante expected revenue is picked.

Consistently with the results found for the estimation experimentation, the evidence from this dataset is that both the exponomial and the Markov chain models perform very well in terms of attainable revenue.

7. PERFORMANCE UNDER BOUNDEDLY RATIONAL PREFERENCES

The ground truths used in the experiments with synthetic instances (Section 5) as well as those using the sushi dataset (Section 6.2) belong to the stochastic preference choice model class (or equivalently, to the ranked list-based or RUM class), which is indeed one of the models tested. In this section, a comparison of the choice models is performed using as a ground-truth a model that goes beyond the RUM class called the *Generalized Stochastic Preference* choice model (GSP) (Berbeglia [12]). An appealing feature of the GSP is that it can explain well-documented behaviors that lie outside the RUM model (such as the *decoy effect* and the *compromise effect*), while at the same time is simple to describe and it encapsulates all the RUM class into it. Under the GSP, each consumer type has an associated list of products without repetitions ℓ jointly with an index $i \in \{1, \ldots, n\}$. When offered an assortment S, the consumer first removes from her list ℓ all the alternatives that *do not* belong to S, and then chooses the alternative that is at the i^{th} position of the updated list ℓ. If all customers types have as their index $i = 1$, the resulting model is a RUM.

Each generated GSP instance contains $n = 10$ products, either 10 or 100 customer types, with 20% of the customers being "non-standard", meaning that their associated index i was greater than 1 [1]. Additionally, 50% of new consumers who always choose not to purchase anything were added. Finally 10 transactions per period were generated, and each period was assigned a random assortment S with $3 \leq |S| \leq 6$. The prices assigned to the products range from 4.69 to 15.99.

Similarly to the experiments with the sushi dataset, Figure 7.1 shows the average out-of-sample soft RMSE performance over 60 randomly generated instances depending on the number of underlying preference list and the volume of training data. The out-of-sample choice predictions (soft RMSE) and revenue performance is compared. While all models perform better when there are 100 lists rather than 10, this improvement is much stronger for the exponomial choice model (consistent with experiments reported in Section 5.2.3). Moreover, the overall results are also similar the experiments carried over the sushi dataset.

In Figure 7.2 the revenue performance is plotted following the same procedure as the one in Section 6.2, except that the full ranking here is kept as the no-choice was always

[1] It is possible to provide reasons to the behaviour of these "non-standard" consumers, such as for example, the lack of attention.

considered as a potential alternative. Again here it is seen that when the number of underlying full rankings is 100, the exponomial shows the best performance. Moreover, the exponomial also performs well even in instances with 10 preferences. One can also see that MNL does not stand out in terms of revenue and has the largest prediction error. One somewhat surprising result is that the rank list-based was not able to perform well even in the case in which there were only 10 preference lists. A plausible explanation is that these instances lie outside the RUM class and as such the column generation algorithm keeps adding (rational) lists without making improvement on the fit.

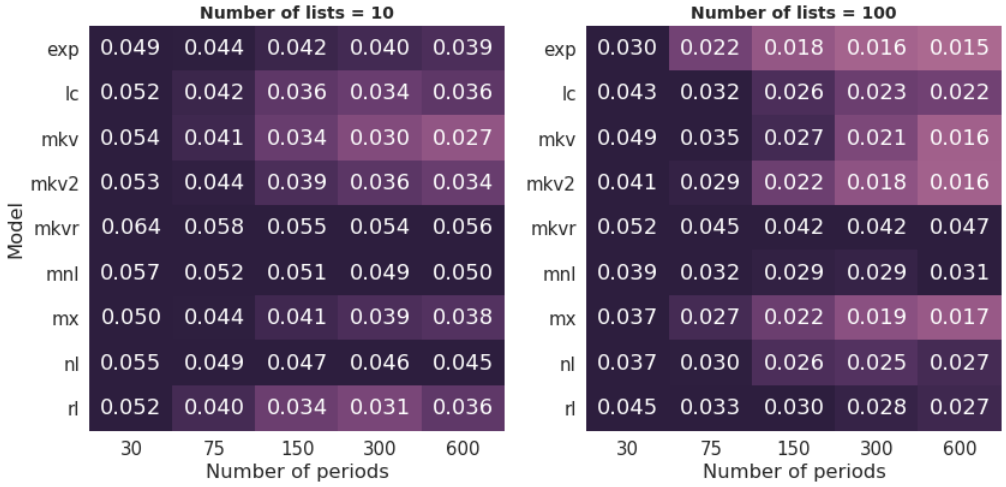


Fig. 7.1: Soft RMSE infinite out of sample for the boundedly rational preferences example. Number of periods in the in-sample (horizontal axis) and choice model (vertical axis). Each value is colored so that the lighter the color, the better the choice model performed.

Fig. 7.2: Revenue assessment from different choice models under boundedly rational preferences. The horizontal axis accounts for the average specification error made by the modeler. The vertical axis accounts for the average percentage of the optimal achievable expected revenues under the ground truth, when both $S^*_{\mathcal{M}}$ and S^* face the ground truth.

8. CONCLUSIONS

Accurate predictions of what consumers will purchase as a function of what they are being offered is fundamental to solve the assortment and price optimization problems. In order to make such predictions, researchers have studied different discrete choice models.

Over the last twenty years, the OM literature has payed significant attention to the use of such discrete choice models to solve different problems which combine operations and consumer behavior (e.g. revenue management in transportation, choice-based demand modeling, estimation and assortment optimization in retail operations), but also in the development (e.g., Markov chain) or the revisiting (e.g., exponomial, ranked list) of alternative choice models.

The performance of the newly proposed discrete choice models is typically assessed in the literature by comparing it to the classic multinomial logit model (MNL). The MNL therefore has been the *de facto* benchmark due to the time it has been around, and its simplicity and popularity. However, it is very rare to find more in depth comparisons between the different choice models in place, and thus, very little is known about their relative predictive performance.

In this work, a systematic and comprehensive empirical study of the most relevant discrete choice models was conducted through an exhaustive set of numerical experiments on synthetic, semi-synthetic and real data. A comparative statistics of the performance quality of the different choice models is provided under a large family of different data environments by varying their historical volume, the structure of the product differentiation, the consistency of the underlying preferences and the product availability. Overall, an analysis over 3,600 different synthetic instances whose ground-truth is the rank list-based model was performed, where each of them was estimated using nine choice models and required overall more than a year of CPU time. Besides these synthetic instances, the performance of the different choice models was compared using (i) synthetic instances based on boundedly rational consumers, (ii) synthetic instances based on the latent class MNL, (iii) synthetic instances based on the nested logit with prices, and (iv) two real datasets: one including 5,000 consumer preferences over ten different sushi types, the other one including more than 3,500 bookings from five hotels. Thanks to this systematic evaluation, the more accurate choice models were characterized depending on some characteristics of the data environment.

The main findings can be summarized as follows. In contexts where the amount of historical data is scarce, the exponomial tended to perform the best (in the aggregate).

Other choice models that performed well in this setting are the mixed logit, the rank list and the nested logit. On the other hand, in settings with a large volume of historical data, the best model overall is the Markov chain, followed by the rank list, the exponomial, and the latent class MNL. These four models turn out to be the best for both unstructured (or random) settings where any product could potentially be selected over any other; and structured settings where there is vertical differentiation between some of the products. It was also analyzed how the consistency of the consumer preferences affect the performance of the different choice models. Most of the models improve their out-of-sample performance when the consistency is low (i.e. the preferences are non-sparse). In settings with high consistency of consumer preferences, the rank list model performed the best, followed by the Markov chain and the latent class MNL. When the consistency of consumer preferences is low, the exponomial was the one that performed the best. It was seconded by the Markov chain and the mixed logit. Following with the real datasets to add robustness to the analysis; in the sushi dataset case, the Markov chain and the exponomial performed the best. In the hotel bookings dataset case, the best models were the exponomial, latent class MNL, Markov chain and ranked-list. Overall these results are consistent with those obtained using synthetic instances. Table 8.1 summarizes the results under two dimensions of the available data: volume and sparsity of preferences. For each cell, the top three choice models are ranked.

The performance of the different choice modes in terms of the achievable revenues was assessed for the sushi dataset and the instances with boundedly rational consumers. As a general conclusion, it is verified that the exponomial model extends its prediction capabilities to an outstanding revenue performance in small data environments. Similarly, the prediction power of the Markov chain model in high volume data environments extends to a noticeable revenue performance therein.

This work has naturally some limitations. First, as in any empirical work, it is always possible to strengthen the results by comparing the models using other real and synthetic datasets. Second, the incorporation of product features (beyond their price) to the synthetic instances, and the analysis of its impact on the models' relative performances is another interesting research direction. Third, the predictive performance of the choice models was assessed in a context where the firm doesn't have or use any specific information about the incoming consumer. However, in an online platform, the seller is able to observe multiple customer attributes (e.g. gender, location, search keyword, etc) and dynamically adjust its choice model (e.g., see Cheung and Simchi-Levi [78]). As a result, the assessment of the different choice models using personalized predictions is an important research direction (e.g., see Jagabathula and Vulcano [79]). Finally, in this

		Degree of consistency of the consumer preferences	
		High (10 lists)	Low (100 lists)
Data Volume	Small	1. Ranked list 2. Exponomial 3. Mixed logit	1. Exponomial 2. Nested logit 3. Mixed logit
	Large	1. Ranked list 2. Markov chain 3. Latent class	1. Exponomial 2. Markov chain 3. Mixed logit

Tab. 8.1: Summary of results under two dimensions of the available data: volume and degree of consistency of the consumer preferences. For each combination the three choice models that performed the best is shown (out-of-sample).

work the relevant choice models coming from the machine learning literature were not assessed. While there has been some recent works in assessing the relative performance of machine learning algorithms and logit models (see, e.g. Zhao et al. [70]), comparing machine learning methods with some of the most performing choice models assessed is an important future research direction.

In summary, the variety of the synthetic instances in terms of number of ground truth types (nonparametric based on the rank list and parametric based on the latent class MNL), consistency of preferences, structure of preferences (completely random preference lists, and preference lists under a rational price structure) and degree of consumers' rationality (full vs. bounded), in addition to two real datasets from different industries (food and hospitality), provide strong support for the final assessments about the most relevant discrete choice models studied in the OM literature.

A Comparative Empirical Study of Discrete Choice Models

Appendix

Appendix 1

IMPLEMENTATION DETAILS

In this section details about the estimation process using least squares is reported, in addition to some details about the computer implementation of the methods.

A1 Least squares estimation

This class of estimators is based on the minimum mean squared error (MSE) criterion -also referred to as *regression estimators*-. Let $F(S)$ be a function that takes a list of T transactions $\{(j_1, S_1), \ldots, (j_T, S_T)\}$ and a specific offer set S, and filters them by leaving only the observations involving that offer set:

$$F(S) = \{(j_t, S_t) : S_t = S, t = 1, \ldots, T\}$$

Now consider the function $f(\mathcal{O}, j)$ that takes a list of observations $\mathcal{O} \subseteq \{(j_1, S_1), \ldots, (j_T, S_T)\}$ and a product j and filters out every transaction which has a product other than j as the observed purchase:

$$f(\mathcal{O}, j) = \{(j, S_t) \in \mathcal{O}\}$$

The sample-based, empirical distribution of the observation (j, S) can be obtained by taking the offer set S and counting the number of times each product was purchased when this set of products was offered across the T observations. The relative frequency of purchases of product j when set S is offered defines the point estimate $\hat{\mathcal{P}}(j|S)$:

$$\hat{\mathcal{P}}(j|S) = \frac{|f(F(S), j)|}{|F(S)|}.$$

The least squares (LS) estimator involves solving a non-linear least squares problem that finds the best suited set of parameters $\boldsymbol{\beta}$ for a given choice model $\mathcal{M}$, taking into account the differences between the true model probabilities $\mathcal{P}_{\mathcal{M}}(j|S, \boldsymbol{\beta})$ and the sample probability estimates $\hat{\mathcal{P}}(j|S)$. The LS estimate of the vector $\boldsymbol{\beta}$ is then the solution to

$$\min_{\boldsymbol{\beta}} \sum_{t=1}^{T} \sum_{j \in S_t \cup \{0\}} \left(\mathcal{P}_{\mathcal{M}}(j|S_t, \boldsymbol{\beta}) - \hat{\mathcal{P}}(j|S_t) \right)^2,$$

where the first sum occurs over all the sets observed in the sample data. This minimization problem can be solved using standard nonlinear optimization methods such as conjugate gradient or quasi-Newton.

A2 Coding of the methods

All models and methods were coded in Python. To find the optimization problem's solutions SciPy **minimize** method was used. In all the estimation methods studied, a time-based stopping criteria was set. Specifically, every method was given 30 minutes to find the best solution they could reach, except for those that met the optimizer's default criteria before reaching the time limit. The experiments were performed by nodes in the cloud provided by the High Performance Computing center in the University of Melbourne (see Meade et al. [74]). This cluster consists of 100 nodes each with 8 cores and 62GB of RAM. All the experiments were divided in batches of instances, these batches were later taken by each estimation method and fed to an specific core within a node as a job, reporting the results when the stopping criterion was met. Each job had a limit of up to 2GB of RAM and could be run in parallel, taking into account that instances and estimation methods were independent.

Appendix 2

MAXIMUM LIKELIHOOD UNDER MARKOV CHAIN: DIRECT OPTIMIZATION VS EM

One of the challenges faced throughout the experimentation was to select the most adequate estimation method for each model. For the Markov chain model, a set of experiments comparing maximum likelihood and EM estimation alternatives was conducted. The EM algorithm implemented is described in Simsek and Topaloglu [42].

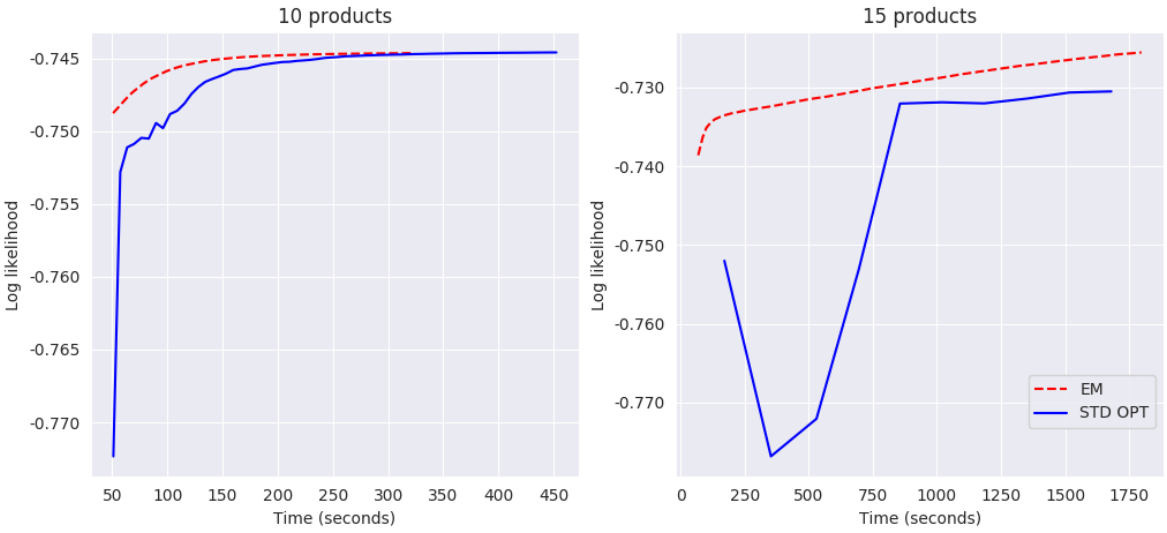


Fig. A1: Maximum likelihood estimation under Markov chain: Direct optimization vs. EM

In Figure A1 it can be seen that as the number of products increases, and given the time limit, the EM method ends up finding a better set of parameters in terms of achievable log-likelihood value. Note that the number of parameters to estimate in the Markov model grows quadratically in n. Although this plot is based on one particular instance, similar results are observed throughout all the set of experiments. Given this evidence, the results of the EM method were reported in the core of this work.

Appendix 3

SUPPLEMENT TO SYNTHETIC DATA EXPERIMENTS

In this section of the appendix complementary results to the ones exhibited in the main body are included.

In Section A1 results for the synthetic data experiments summarized in Section 5.2 are presented. In addition to the soft RMSE, the mean absolute percentage error (MAPE) is reported. Given a ground truth model θ and an estimated model ϕ, the MAPE is computed as follows:

$$MAPE(\mathcal{P}_\theta, \mathcal{P}_\phi) = \frac{1}{\sum_{S\subseteq\mathcal{N}}(|S|+1)} \sum_{S\subseteq\mathcal{N}} \sum_{j\in S\cup\{0\}} \frac{|\mathcal{P}_\theta(j|S) - \mathcal{P}_\phi(j|S)|}{\mathcal{P}_\theta(j|S)}.$$

Note that the ratio in the formula is well defined since a pair (j, S) could appear in the in-sample data as long as $P_\theta(j|S) > 0$. There results are based on maximum likelihood estimation. Figures A1 and A2 show soft RMSE and MAPE results, respectively, for the case of 10 products and unstructured instances, followed by soft RMSE and MAPE results on structured instances with 15 products in Figures A3 and A4, respectively. Their counterpart results under least square estimation are presented in Section A2, in Figures A5 through A8. It is worth mentioning that in a few cases, and somewhat surprising, the performance of the different models might deteriorate as the number of periods in the training set increases from 300 to 600. This is not an anomaly of the model or the estimation method but rather a consequence of dealing with a limited computational budget (in this case, 30 minutes). Since the direct maximization of the log-likelihood function (or the minimization of the least square function) imply the calculation of the gradient across 6,000 transactions, it becomes computationally costly to perform each iteration and therefore, a small number of them could be executed. In those cases, a careful balance between the quality of the gradient and the number of iterations may mitigate this effect.

In Section A3 complementary results about the effect of the consistency of the underlying preferences is presented, extending the comparison between a ground truth with 10 and 100 lists (as presented in Section 5.2.3) to the comparison between a ground truth with 2 and 100 lists.

The synthetic data results reported in the main body (Section 5.2) are based on a rank list-based ground truth. In Section A4 the out-of-sample performance of the different discrete choice models studied is reported for a series of experiments in which the ground-truth model is the latent class MNL.

This supplement is closed with Section A5, where a ground-truth in which consumers are sensitive to product prices is considered.

A1 Maximum likelihood

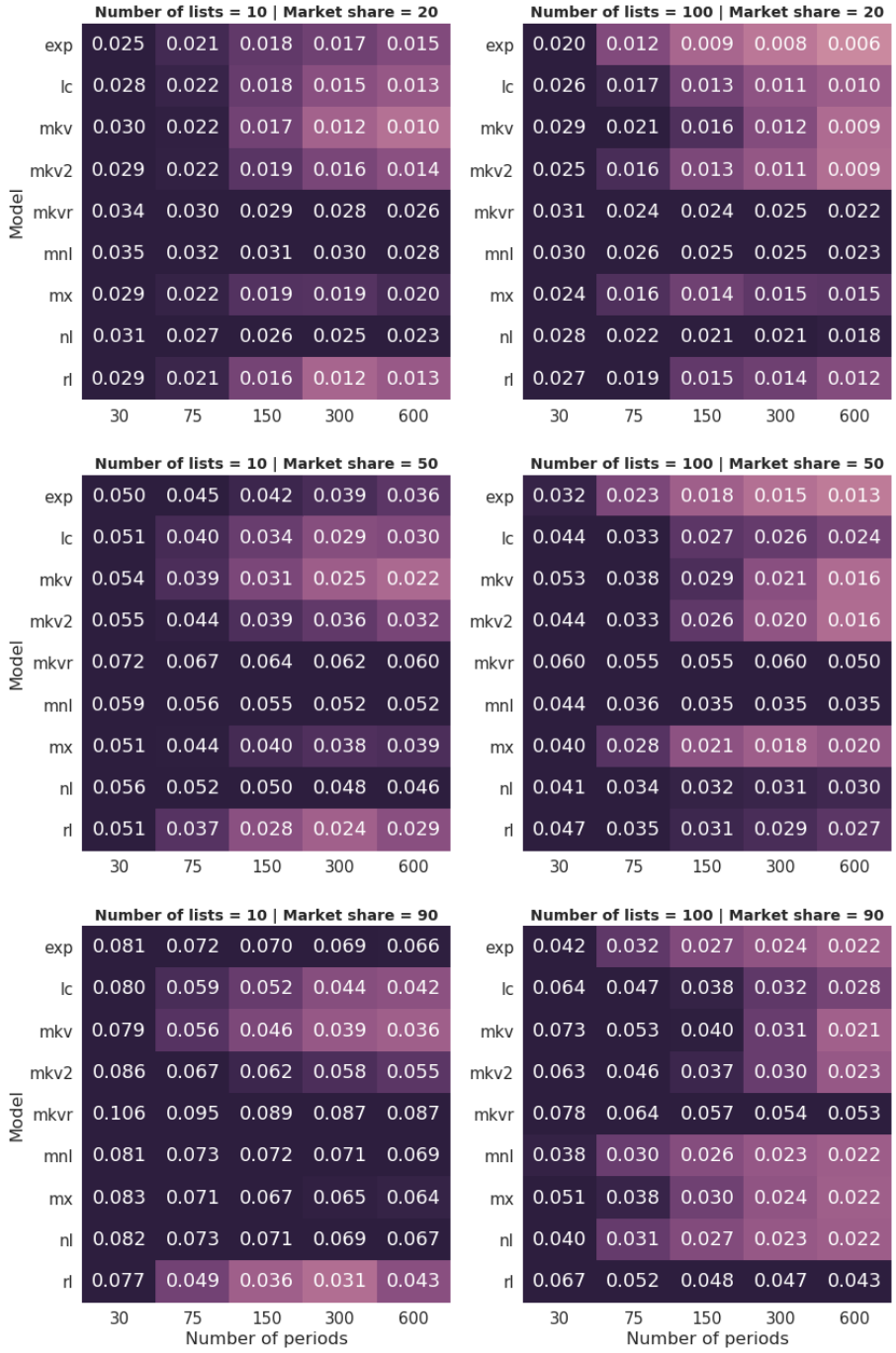


Fig. A1: Results with 10 products and random instances. Each row stands for a model, and each column represents a number of periods in the training data set. For each model and number of periods, the corresponding soft RMSE is reported.

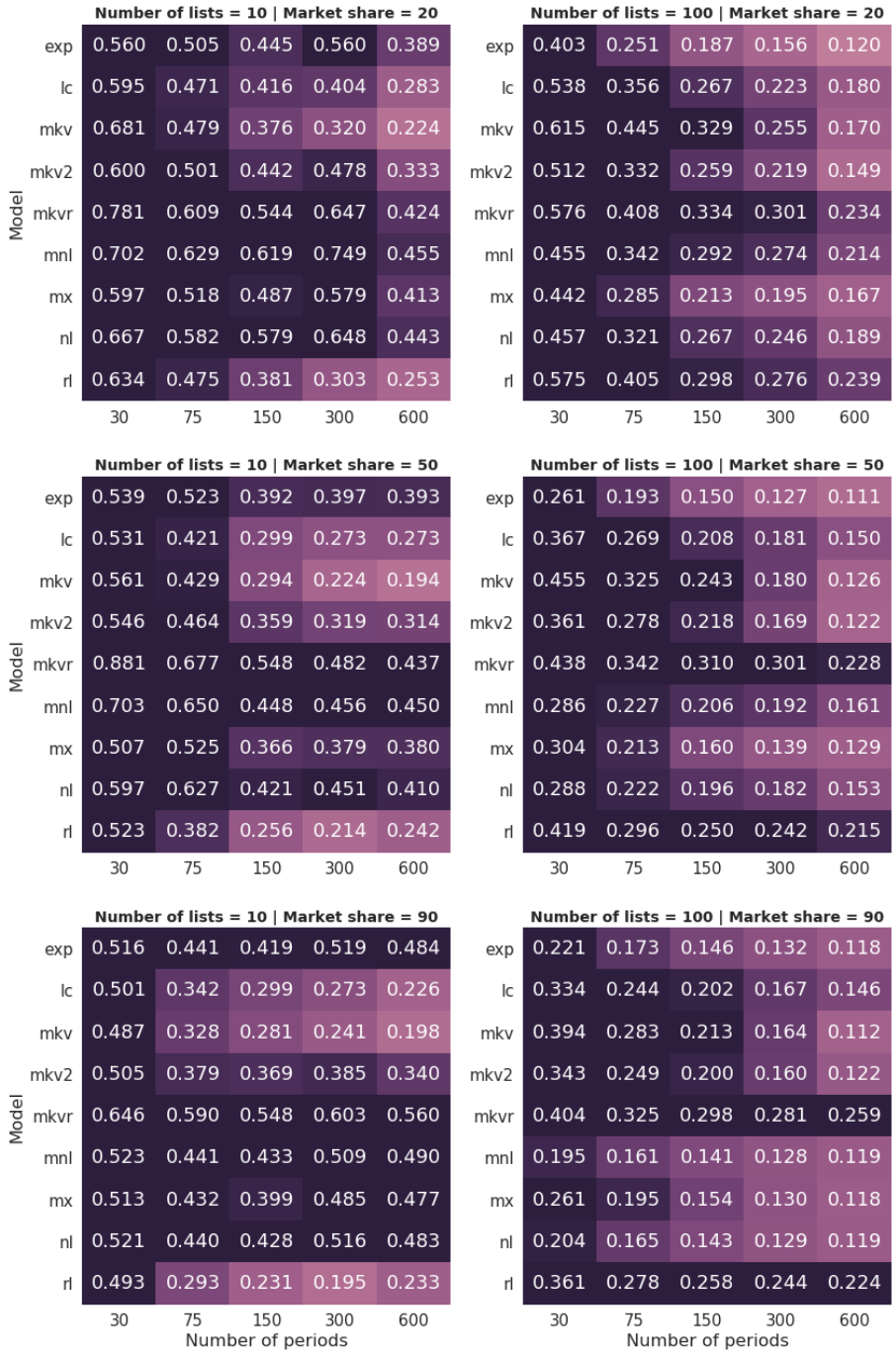


Fig. A2: Results with 10 products and random instances. Each row stands for a model, and each column represents a number of periods in the training data set. For each model and number of periods, the corresponding MAPE is reported.

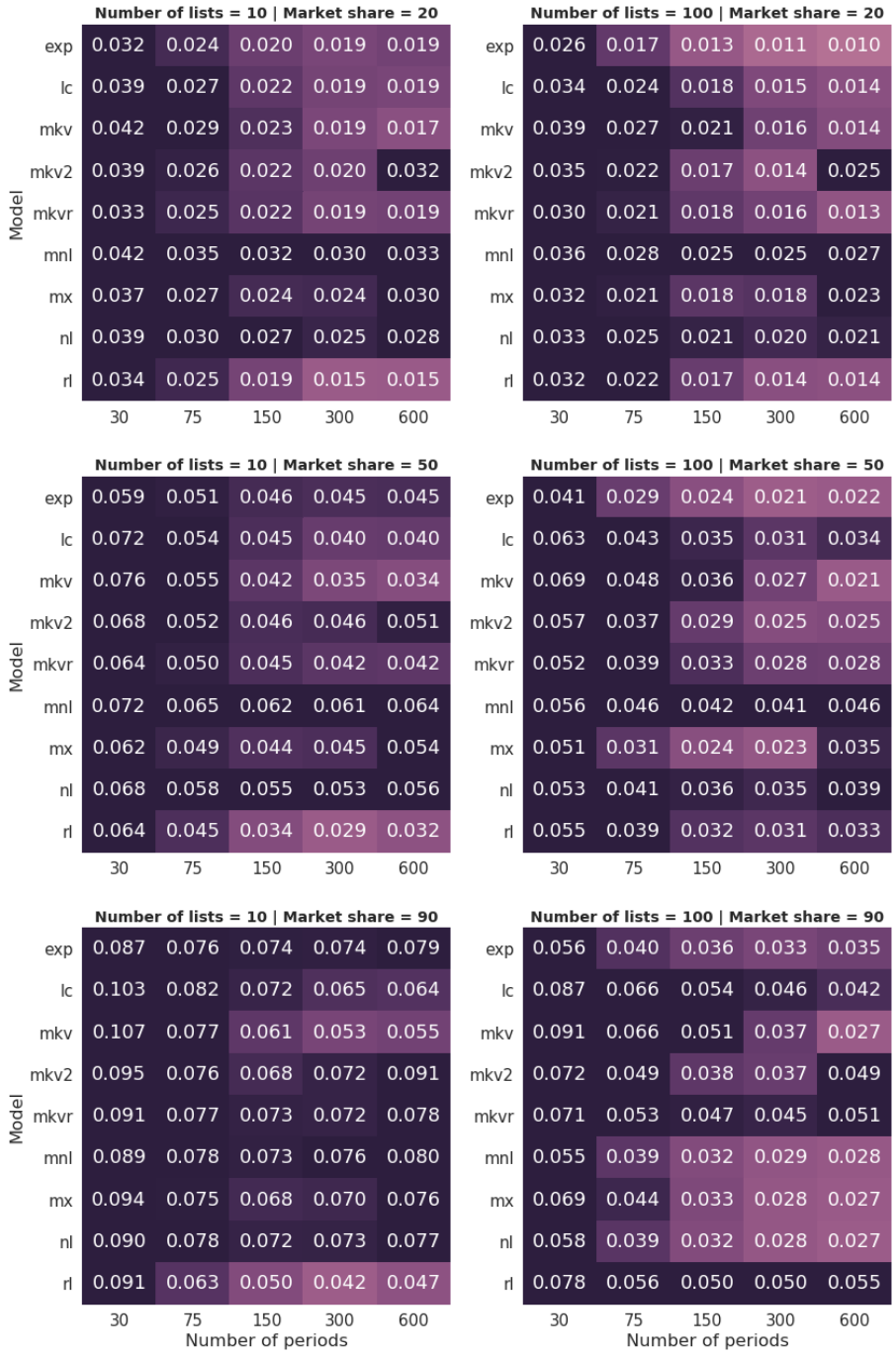


Fig. A3: Results with 15 products and structured instances. Each row stands for a model, and each column represents a number of periods in the training data set. For each model and number of periods, the corresponding soft RMSE is reported.

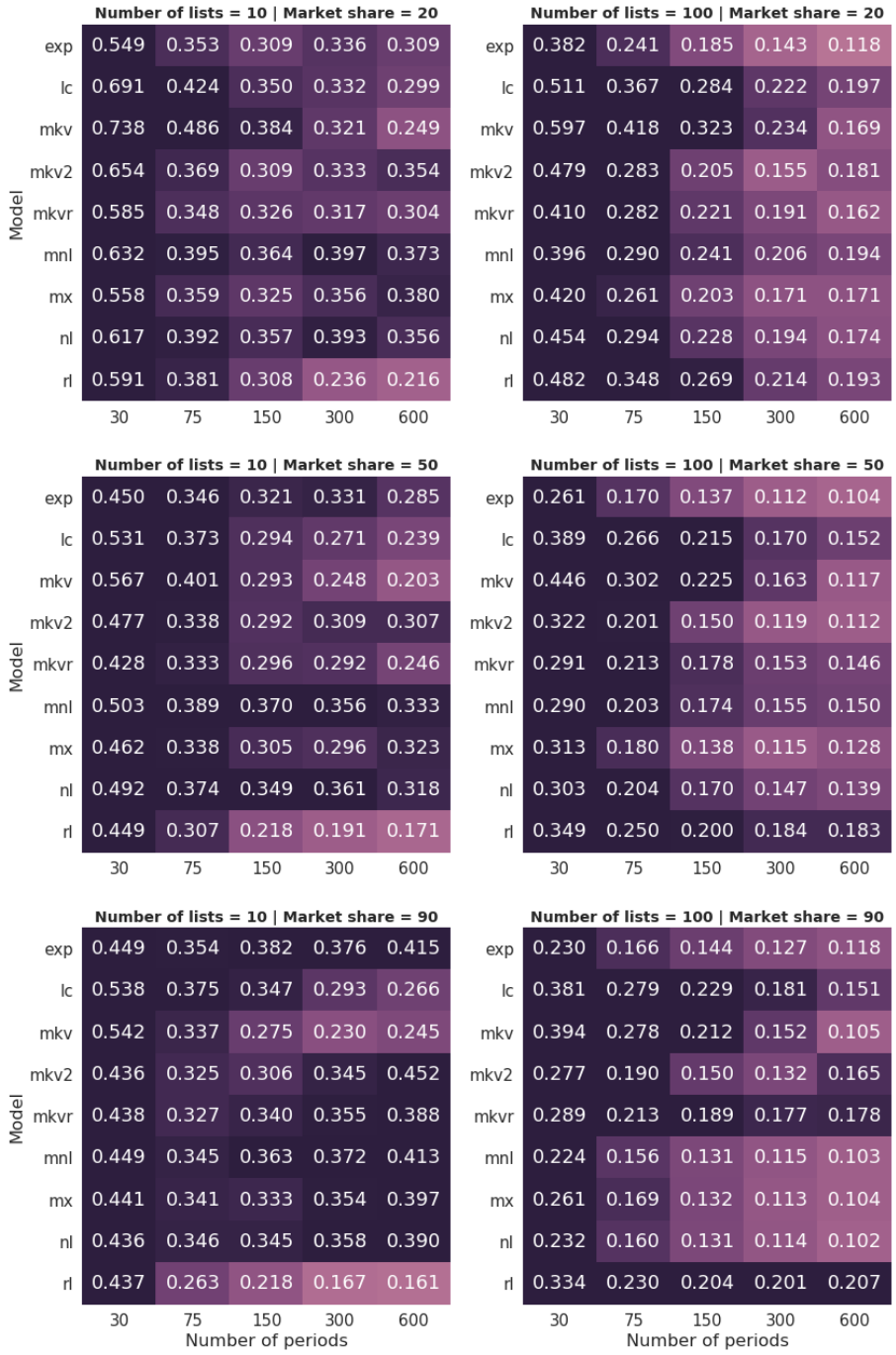


Fig. A4: Results with 15 products and structured instances. Each row stands for a model, and each column represents a number of periods in the training data set. For each model and number of periods, the corresponding MAPE is reported.

A2 Least squares

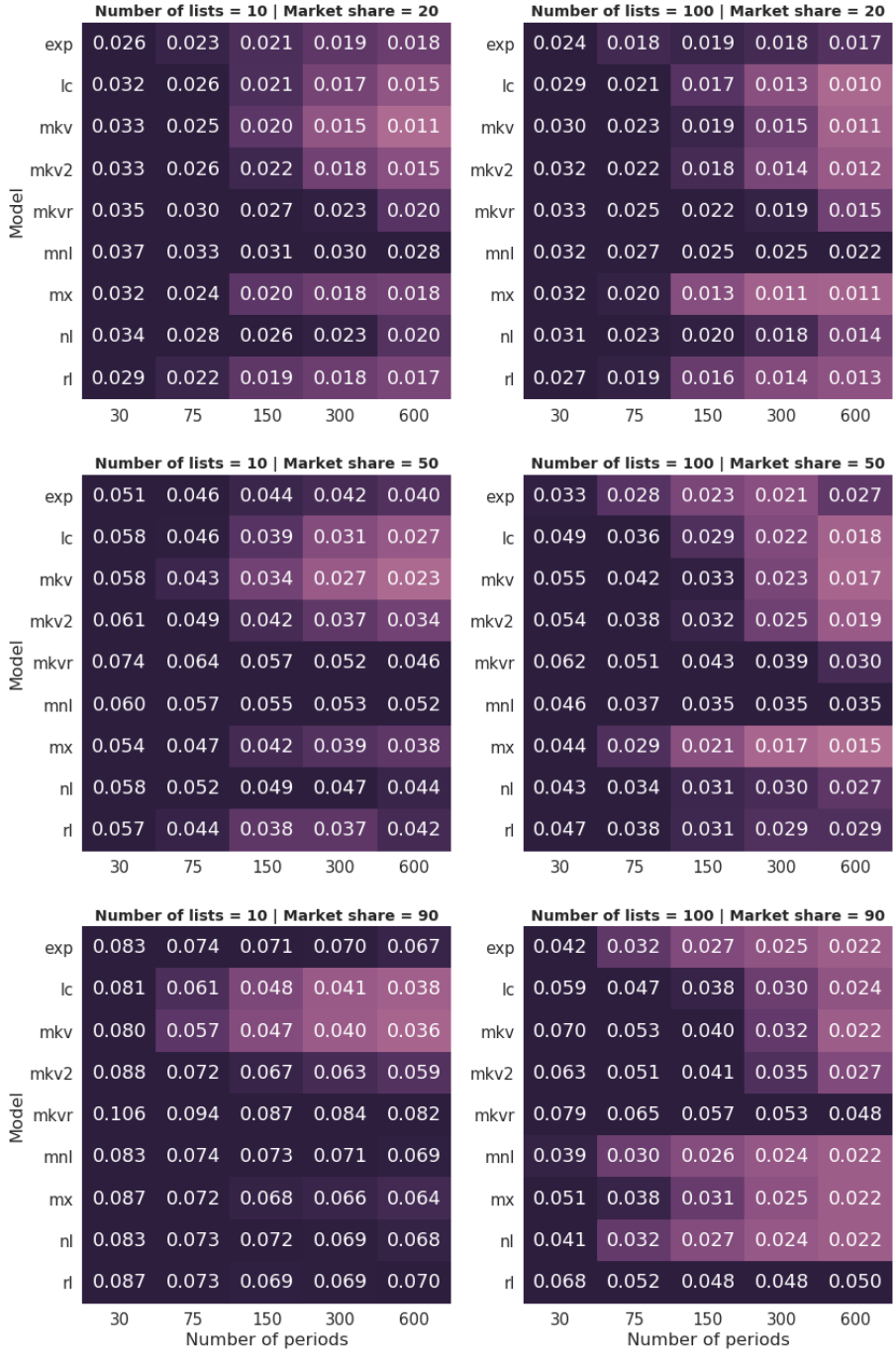


Fig. A5: Results with 10 products and random instances. Each row stands for a model, and each column represents a number of periods in the training data set. For each model and number of periods, the corresponding soft RMSE is reported.

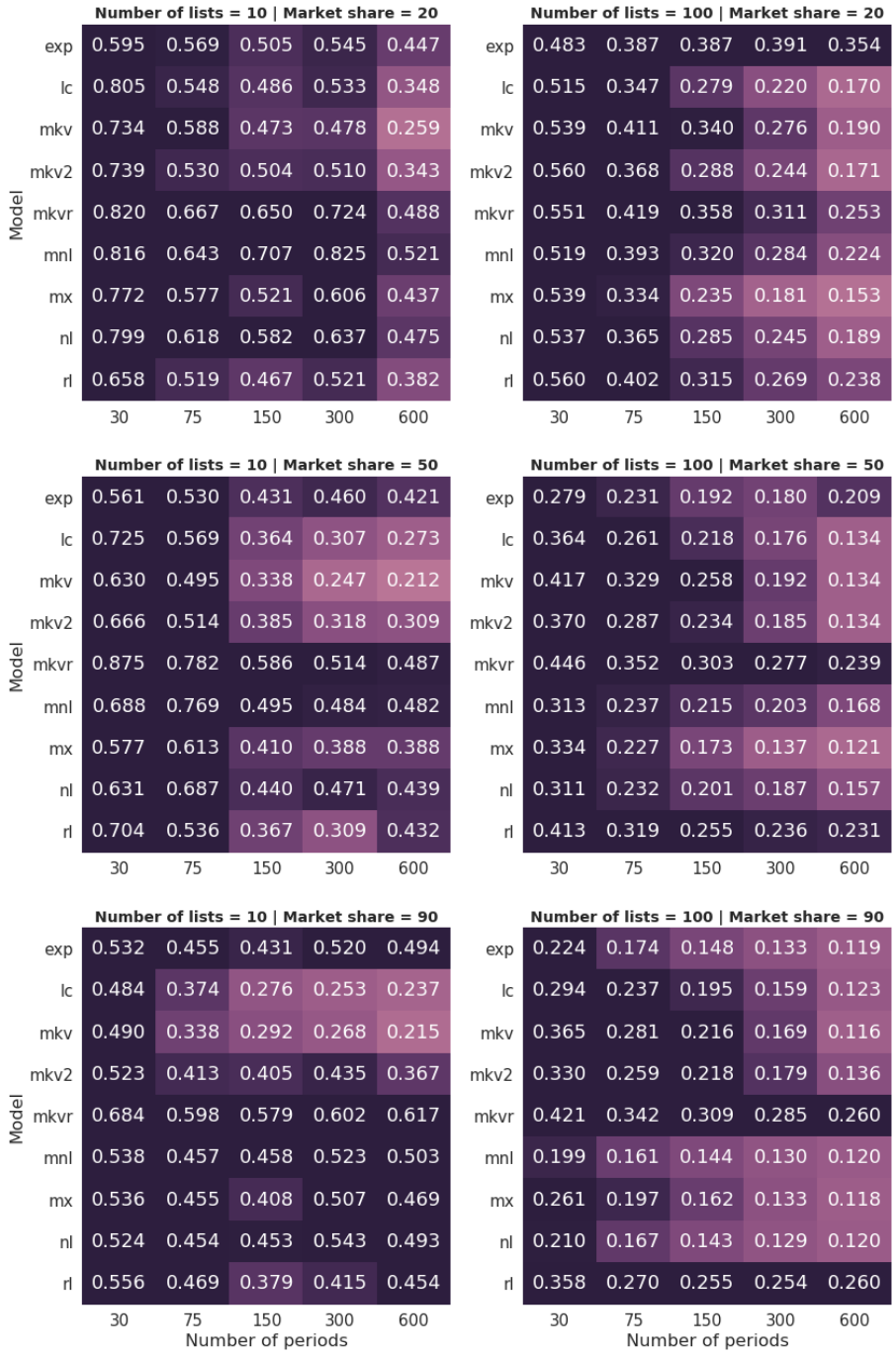


Fig. A6: Results with 10 products and random instances. Each row stands for a model, and each column represents a number of periods in the training data set. For each model and number of periods, the corresponding MAPE is reported.

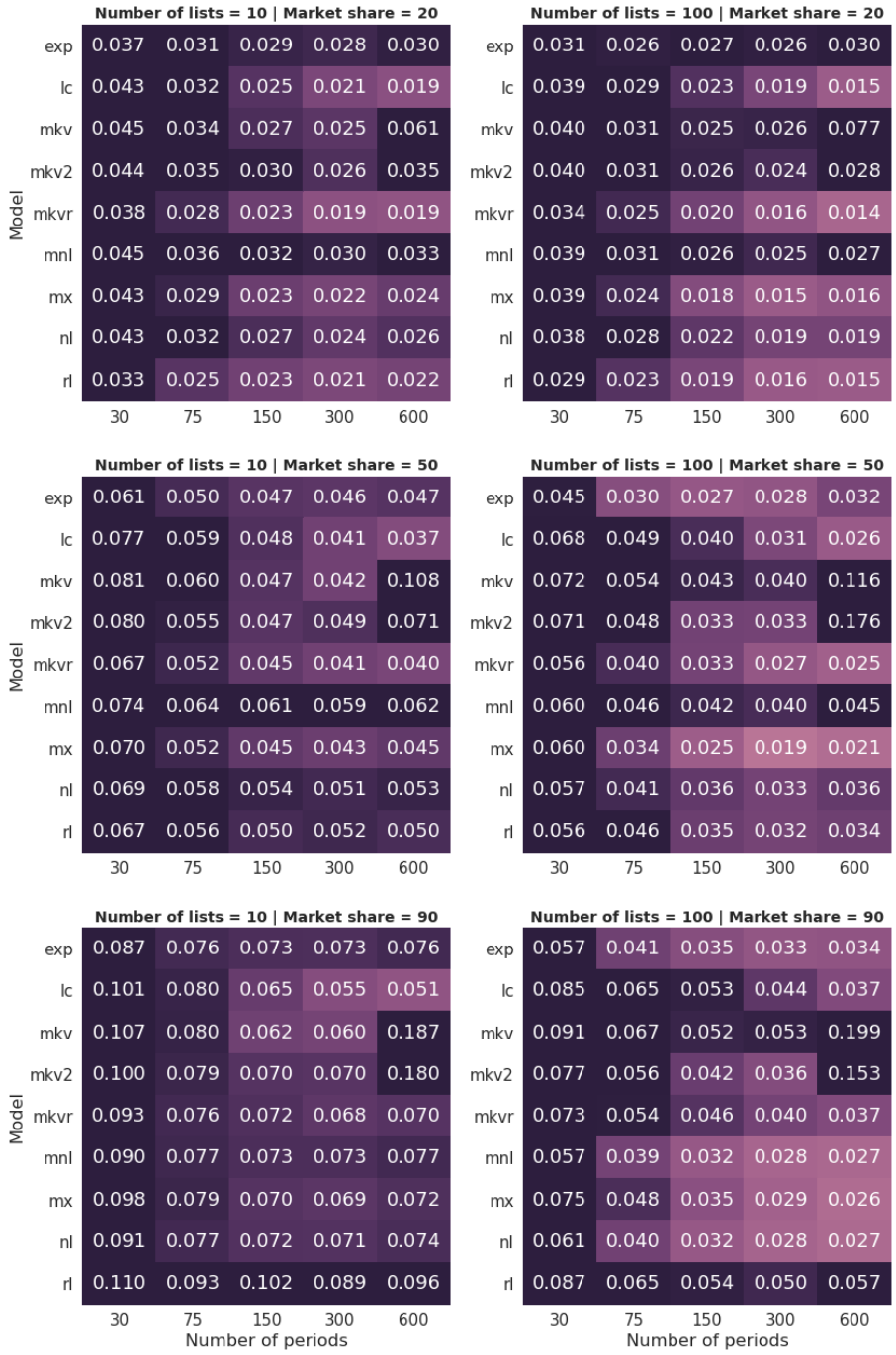


Fig. A7: Results with 15 products and structured instances. Each row stands for a model, and each column represents a number of periods in the training data set. For each model and number of periods, the corresponding soft RMSE is reported.

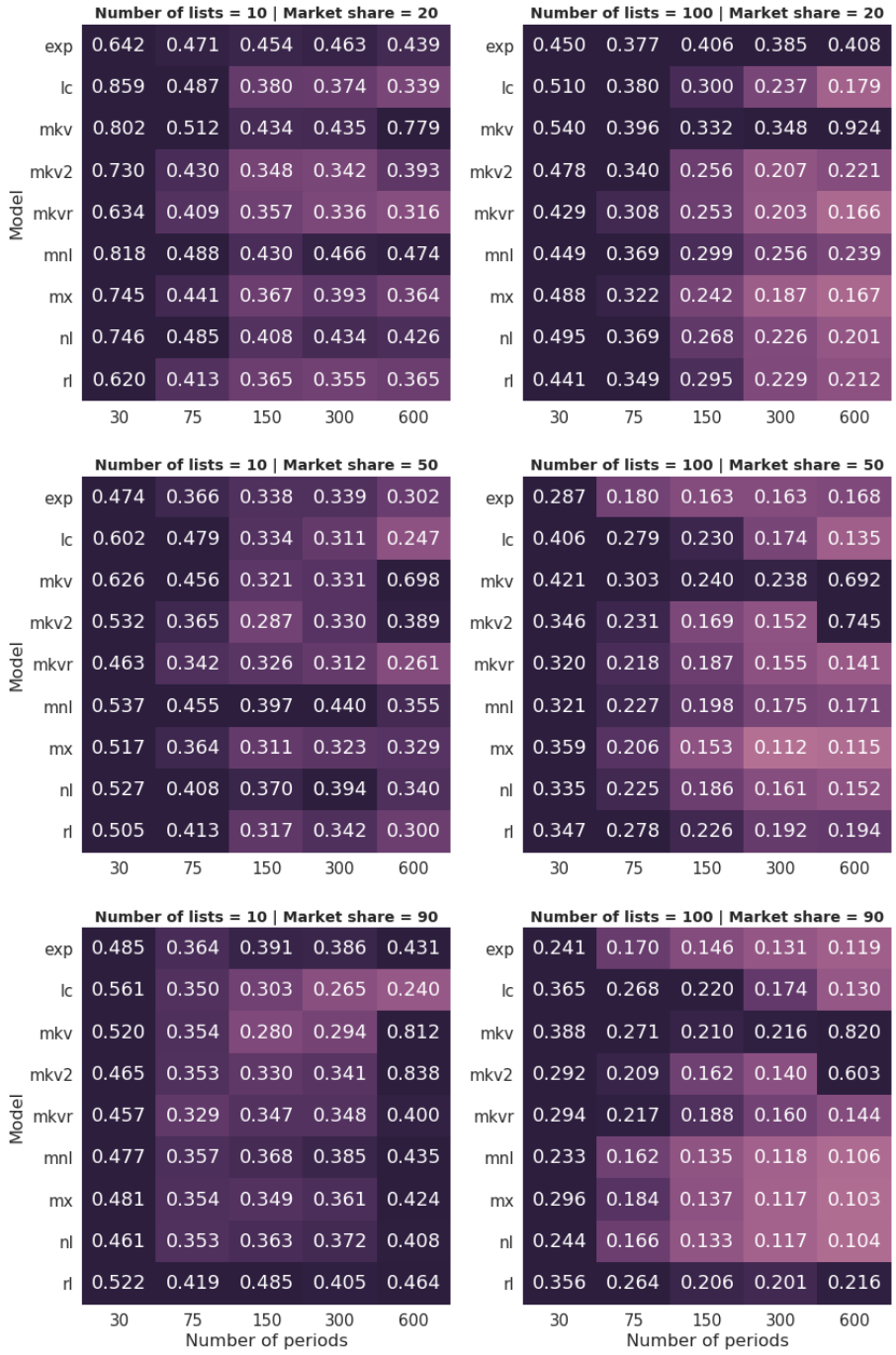


Fig. A8: Results with 15 products and structured instances. Each row stands for a model, and each column represents a number of periods in the training data set. For each model and number of periods, the corresponding MAPE is reported.

As shown in Tables A5 and A7, the observed patterns are generally similar to Tables A1 and A3, respectively. A clear difference though is the poorer performance of the Markov chain model when there are 600 periods over the structured instances, when the solution

of the related least squares problem becomes more computationally intensive.

A3 Consistency of preferences: a comparison between 2 lists and 100 lists

In this section the relative performance of the different choice models is checked on instances with a ground-truth of only two preference lists versus instances based on 100 preference lists (see Figure A9). Following the performance pattern between 10 and 100 lists, it is observed that the ranked list model performs even better over 2 preference lists compared to the 100 lists case. On the other hand, the exponomial continues to worsen in terms of relative performance from 100 to 2 lists.

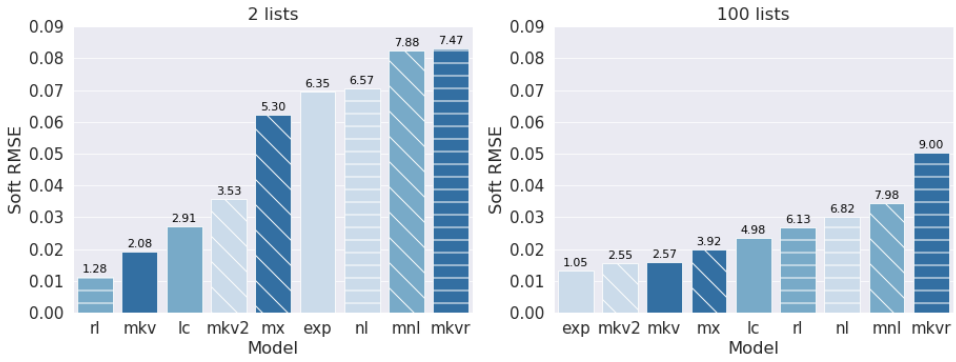


Fig. A9: Impact of the consistency of the underlying preferences. Average soft RMSE in instances with 2 preference lists (left) and 100 preference lists (right). In-sample size is set to 6,000 transactions. The ranked list and Markov chain models achieve the best out-of-sample results in the 2-list scenarios.

A4 Latent class ground truth

In this section the out-of-sample performance of the discrete choice models studied is reported for a series of experiments in which the ground-truth model is the latent-class MNL model. Then, these results are compared with those obtained when the ground-truth is the rank list model (Section 5.2).

The latent class MNL model, similar to the mixed logit model, is a very general choice model that is contained in the RUM class. In a seminal work, McFadden and Train [36] proved that for every $\epsilon > 0$, any random utility model can be ϵ-approximated* by a latent class MNL model. One limitation of this important result is that such approximation may require an exponential number of classes. Recently, Chierichetti et al. [80] sharpened

* Here this means that a choice model $\mathcal{P}'$ is an ϵ-approximation of a choice model $\mathcal{P}$ if $|\mathcal{P}'(i,S) - \mathcal{P}(i,S)| \leq \epsilon$ for every $i \in S \cup \{0\}, S \subseteq \mathcal{N}$.

McFadden and Train's result by showing that every RUM can be approximated by a latent-class MNL with a polynomial number of classes.

The method used to generate the latent class MNL instances is based on Chierichetti et al. [80] and it is as follows. To generate the nominal utility V_{ic} for each alternative i under class c, first a permutation σ_c of the $N+1$ alternatives (including the no-purchase) that is associated to class c is randomly created. Second,

$$V_{ic} = \epsilon^{\sigma_c^{-1}(i)}$$

is assigned for every $i \in \mathcal{N} \cup \{0\}$ where $\sigma_c^{-1}(i)$ denotes the position of alternative i in the permutation σ_c. By varying the value of ϵ between zero and one similarity of the utilities of the alternatives for the consumers of a given class can be adjusted, where ϵ close to zero implies very different utilities, and ϵ close to one implies similar utilities.

Instances with $N = 10$ products and with either $k = 10$ or $k = 100$ classes were created. For each of those two types, instances with three different values for ϵ were generated: 0.5, 0.1, and 0.001. Similarly to the instance generation procedure for the ranked-list, instances with 30, 75, 150, 300 and 600 periods were constructed, where each period consists of 10 transactions. The assortment S associated to each period satisfies $3 \leq |S| \leq 6$. In addition, a market share of 50% is set by enforcing fifty percent of the classes to have the no-purchase alternative (product 0) as their most valuable option (i.e. $\sigma_c^{-1}(0) = 1$). Finally, for each combination of the classes k, ϵ and number of periods, 60 instances were randomly generated. The average soft-RMSE for the different choice models is shown in Figure A10.

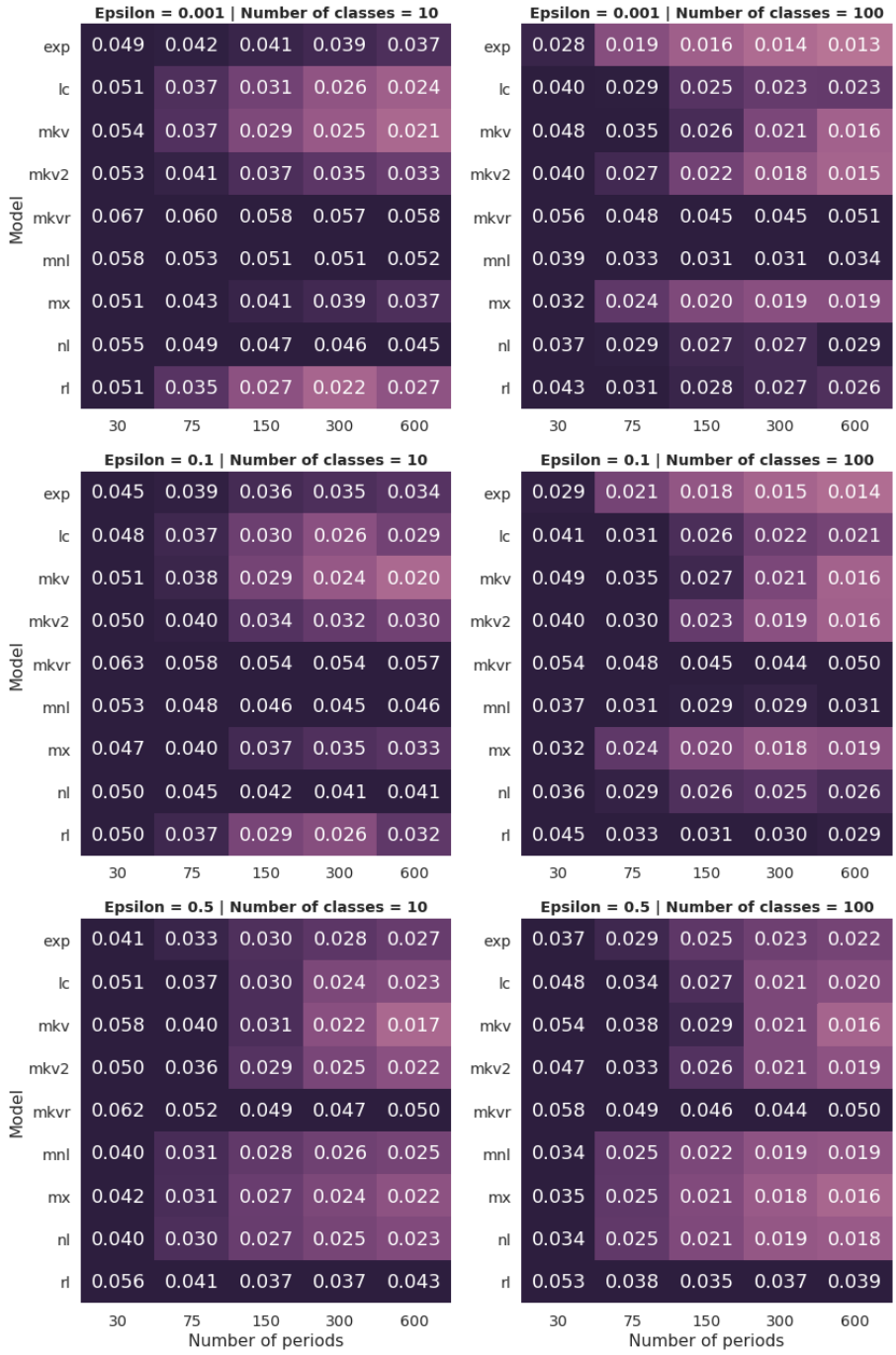


Fig. A10: Average soft-RMSE when the ground-truth comes from the latent class MNL model.

When the value of ϵ tends to zero, the product utilities associated to each consumer class decrease exponentially following the order of the associated ranking. Moreover, when ϵ tends to zero, the resulting model tends to a stochastic preference choice model. The top two tables from Figure A10 have $\epsilon = 0.001$ it is observed that the results are almost the same as those obtained when the ground truth is the stochastic preference choice model.

Specifically, one should compare these results with the results shown in Figure A1 when the market share is 50 percent (the two tables in the middle in that figure). Observe that on the other extreme, when the value of ϵ approaches 1, the instances become "smoother" and get closer to a (uniform) random choice model in which the probability of choosing $i \in S \cup \{0\}$ approaches $\frac{1}{|S|+1}$. From the results in Figure A10 it can be seen that as ϵ increases, the MNL and its extensions (latent class MNL, mixed MNL and nested logit MNL) improve in terms of soft-RMSE. It is also interesting to see that the performance of the exponomial for the different ϵ values differs dramatically depending on whether there are 10 classes or 100 classes. This is consistent to that observed when the ground truth was the stochastic preference model.

The two models that perform poorly as ϵ increases are the reduced Markov chain and the ranked-list. The reasons that explain this behavior are different. In case of the reduced Markov chain, it is matter of its superimposed structure that prevents it from adjusting to a uniform random choice as some products are enforced to be preferable over others. In the case of the ranked-list, it is a matter of the stopping criteria used that halts the column generation procedure with a few dozens of preference lists. If a more exhaustive range of lists was considered, close to the total support of cardinality $O(n!)$, the performance would improve substantially.

A5 A ground-truth with prices

In all of the previous synthetic experiments, product attributes were not used to estimate the demand for the different scenarios. In this section, a ground-truth in which consumers are sensitive to product prices will be considered. The different models will be trained as usual based on a set of transactions but now the products prices in the in-sample transactions will vary. The test-phase will run similarly to the previous experiments in that it only contains assortments that are not present in the in-sample. Moreover, the prices in the out-of-sample will also differ from those in the in-sample.

To bring additional variety to the choice models used as grounds-truths, for this experiment a series of ground-truth instances based on a nested logit MNL were constructed. Some of the parameters come from a series of experiments carried out by P. Rusmevichientong and Shmoys [81] that estimate DVD's sales at Amazon. P. Rusmevichientong and Shmoys [81] considered that consumers followed an MNL model where the nominal utility U_i, of product i, is as follows:

$$u_i = b_2 v_i - b_1 p_i + b_0,$$

where $v_i \geq 0$ is the number of up-votes that product i received at Amazon, and p_i is

its price. The values b_2, b_1 and b_0 are the same for all the products and were estimated in P. Rusmevichientong and Shmoys [81] using on all the transactions. Those values are $b_0 = -4.31$, $b_1 = 0.038$, and, $b_2 = 3.54 * 10^{-5}$. For the experiments $N = 10$ products were selected by taking the top 10 products with highest sales out of the 15 products used by Farias et al. [59].

Based on the above utilities, a series of 2-class nested logit instances were constructed as follows: The first nest N_1 consists of the first five products whereas the second nest consisted of the remaining ones (i.e. $N_1 = \{1, 2, 3, 4, 5\}$ and $N_2 = \{6, 7, 8, 9, 10\}$). Under the nested logit model, each nest i has an associated parameter λ_i that encodes the degree of the dissimilarity of the products in the nest. Here $\lambda_1 = \lambda_2 = 0.7$ is set, and the nominal utility for product $i \in N_j$ with $j \in \{1, 2\}$ is set to be:

$$u'_i = b_2 v_i - \alpha_j \cdot b_1 p'_i + b_0,$$

where p'_i is the product price, and the price sensitivities are $\alpha_1 = 0.5$ and $\alpha_2 = 1.5$. Thus, consumers have higher price sensitivity for products in the second nest (the values v_i, $b0$, b_1, and b_2 were not modified).

For each possible value of the number of periods $T \in \{30, 75, 150, 300, 600\}$, 60 instances were generated by varying the offer sets and the prices. Specifically, for each period an assortment S was randomly generated with cardinality between 3 and 6. At each period, the price p'_i is randomized for a product $i \in S$ based on the product full price p_i extracted from [81] as follows: full price with 50% chance ($p'_i = p_i$);25% discount with 25% chance ($p'_i = 0.75 p_i$); and 50% discount with 25% chance ($p'_i = 0.5 p_i$). Similarly to the other instance generation processes, 10 transactions per period were created and the market share was set to 50 percent.

The out-of-sample assortments contained all the possible assortments except those that appeared in the in-sample. For each instance, the price p'_i used for product i across all the out-of-sample assortments is the weighted average price for product i across all the in-sample transactions.

The estimation procedures were implemented accounting for price sensitivity for the following four choice models: MNL, nested logit MNL, latent-Class MNL and exponomial. While there are some interesting recent approaches to modify the ranked-list[82] and the Markov chain models[83] to incorporate pricing information, those methodologies are non-standard since prices do not directly modify product utilities (as utilities are not explicit in those models). As a result, a methodology for doing pricing in those models has not yet been established in the research community and it is an important ongoing research topic.

The average soft-RMSE for the different choice models tested is shown in Figure A11.

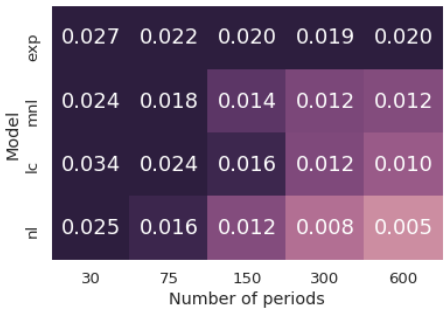


Fig. A11: Average soft-RMSE when the ground-truth comes from the nested logit MNL model.

It can be observed that the prediction performance of the four models improve considerably when the number of periods increased from 30 to 75. Moreover, at 75 periods, the four models have similar performance. However, when the models using an in-sample of 150 periods and more are compared, a considerable performance deterioration can be seen in the exponomial relative to the performance of the MNL and its extensions (latent class and nested logit MNL). This is not very surprising since the ground truth in this experiment follows a nested logit model with 2 nests. Although the structure of a 2-nest nested logit model differs from the standard MNL and the LC-MNL, it can be seen that those three models have much more in common in comparison with the exponomial choice model. When the number of transactions is the largest ($T = 600$), it can be seen that the LC-MNL surpasses the subsumed MNL as it is able to learn more from in-sample transactions. Finally, observe that the NL ranked first in terms of out-of-sample predictions. This is something expected as the ground truth in these experiments is also a nested logit.

BIBLIOGRAPHY

[1] T. Yunes, D. Napolitano, A. Scheller-Wolf, and S. Tayur. Building efficient product portfolios at John Deere and company. *Operations Research*, 55(4):615–629, 2007.

[2] J. Lee, V. Gaur, S. Muthulingam, and G. Fisher. Stockout-based substitution and inventory planning in textbook retailing. *M&SOM*, 18(1):104—121, 2016.

[3] G. Vulcano, G. van Ryzin, and W. Chaar. Choice-based revenue management: An empirical study of estimation and optimization. *M&SOM*, 12(3):371–392, 2010.

[4] J. Dai, W. Ding, A. Kleywegt, X. Wang, and Y. Zhang. Choice based revenue management for parallel flights. Working paper, ISyE, Georgia Institute of Technology, GA, available at SSRN 2404193, 2014.

[5] A. Musalem, M. Olivares, E. Bradlow, C. Terwiesch, and D. Corsten. Structural estimation of the effect of out-of-stocks. *Management Science*, 56(7):1180–1197, 2010.

[6] G. Kök and M. Fisher. Demand estimation and assortment optimization under substitution: Methodology and application. *Operations Research*, 55(6):1001–1021, 2007.

[7] M. Cohen, Z. Leung, G. Perakis, K. Panchamgam, and A. Smith. The impact of linear optimization on promotion planning. *Operations Research*, 65(2):446–468, 2017.

[8] K. Train. *Discrete choice methods with simulation*. Cambridge University Press, New York, NY, second edition, 2009.

[9] G. Berbeglia and G. Joret. Assortment optimisation under a general discrete choice model: A tight analysis of revenue-ordered assortments. *Algorithmica*, 82(4):681–720, 2020.

[10] J. Huber, J. Payne, and C. Puto. Adding asymmetrically dominated alternatives: Violations of regularity and the similarity hypothesis. *Journal of Consumer Research*, 9:90–98, 1982.

[11] F. Echenique, K. Saito, and G. Tserenjigmid. The perception-adjusted Luce model. *Mathematical Social Sciences*, pages 67–76, 2018.

[12] G. Berbeglia. The generalized stochastic preference choice model. Working paper, Melbourne Business School, Australia, arXiv preprint arXiv:1803.04244, 2018.

[13] F. Echenique and K. Saito. General Luce model. *Economic Theory*, 68:811–826, 2019.

[14] A K Strauss, R Klein, and C Steinhardt. A review of choice-based revenue management: Theory and methods. *European Journal of Operational Research*, 271(2): 375–387, 2018.

[15] C. Queenan, M. Ferguson, J. Higbie, and R. Kapoor. A comparison of unconstraining methods to improve revenue management systems. *Production and Operations Management*, 16:729–746, 2007.

[16] D. Luce. *Individual Choice Behavior*. John Wiley and Sons, 1965.

[17] P. Guadagni and J. Little. A logit model of brand choice calibrated on scanner data. *Marketing Science*, 2:203–238, 1983.

[18] R. Ratliff, B. Rao, C. Narayan, and K. Yellepeddi. A multi-flight recapture heuristic for estimating unconstrained demand from airline bookings. *Journal of Revenue and Pricing Management*, 7:153–171, 2008.

[19] J. Newman, M. Ferguson, L. Garrow, and T. Jacobs. Estimation of choice-based models using sales data from a single firm. *M&SOM*, 16:184–197, 2014.

[20] K. T. Talluri and G. J. van Ryzin. Revenue management under a general discrete choice model of consumer behavior. *Management Science*, 50:15–33, 2004.

[21] Z. Li. A single-period assortment optimization model. *Production and Operations Management*, 16(3):369–380, 2007.

[22] S. Anderson and A. de Palma. Multi-product firms: A nested logit multi-product firms: A nested logit approach. *Journal of Industrial Economics*, 40:261—276, 1992.

[23] W. Hopp and X. Xu. Product line selection and pricing with modularity in design. *M&SOM*, 7(3):172–187, 2005.

[24] C. Daganzo. *Multinomial Probit: The Theory and Its Application to Demand Forecasting*. Academic Press, New York, NY, 1979.

[25] A. Alptekinoğlu and J. Semple. The exponomial choice model: A new alternative for assortment and price optimization. *Operations Research*, 64(1):79–93, 2016.

[26] A. Aouad, J. Feldman, and D. Segev. The exponomial choice model: Algorithmic frameworks for assortment optimization and data-driven estimation case studies. *Available at SSRN 3192068*, 2018.

[27] L. Subramanian S. Jagabathula and A. Venkataraman. A conditional gradient approach for nonparametric estimation of mixing distributions. *Management Science*, 2020.

[28] Q. Liu and G. van Ryzin. On the choice-based linear programming model for network revenue management. *M&SOM*, 10(2):288–310, 2008.

[29] I. Mendez-Diaz J. Miranda-Bront and G. Vulcano. A column generation algorithm for choice-based network revenue management. *Operations Research*, 57(3):769–784, 2009.

[30] J. Feldman and H. Topaloglu. Bounding optimal expected revenues for assortment optimization under mixtures of multinomial logits. *Production and Operations Management*, 24(10):1598–1620, 2015.

[31] P. Rusmevichientong, D. Shmoys, C. Tong, and H. Topaloglu. Assortment optimization under the multinomial logit model with random choice parameters. *Production and Operations Management*, 23:2023–2039, 2014.

[32] J. Davis, G. Gallego, and H. Topaloglu. Assortment optimization under variants of the nested logit model. *Operations Research*, 62:250–273, 2014.

[33] D. Brownstone and K. Small. Efficient estimation of nested logit models. *Journal of Business & Economic Statistics*, 7(1):67–74, 1989.

[34] G. Gallego and H. Topaloglu. Constrained assortment optimization for the nested logit model. *Management Science*, 60(10):2583–2601, 2014.

[35] G. Li, P. Rusmevichientong, and H. Topaloglu. The d-level nested logit model: Assortment and price optimization problems. *Operations Research*, 63(2):325–342, 2015.

[36] D. McFadden and K. Train. Mixed mnl models for discrete response. *Journal of Applied Econometrics*, 15:447–470, 2000.

[37] J. Blanchet, G. Gallego, and V. Goyal. A Markov chain approximation to choice modeling. *Operations Research*, 64(4):886–905, 2016.

[38] G. Berbeglia. Discrete choice models based on random walks. *Operations Research Letters*, 44(2):234–237, 2016.

[39] A. Gupta and D. Hsu. Parameter identification in Markov chain choice models. In *Twenty-Eigth International Conference on Algorithmic Learning Theory*, 2017.

[40] J. Feldman and H. Topaloglu. Revenue management under the Markov chain choice model. *Operations Research*, 65(5):1322–1342, 2017.

[41] A. Désir, V. Goyal, D. Segev, and C. Ye. Capacity constrained assortment optimization under the Markov chain model. *Management Science*, 66(2):698–721, 2020.

[42] A. S. Şimşek and H. Topaloglu. An expectation-maximization algorithm to estimate the parameters of the Markov chain choice model. *Operations Research*, 66(3):748–760, 2018.

[43] H. Block and J. Marschak. Random orderings and stochastic theories of responses. *Contributions to probability and statistics*, 2:97–132, 1960.

[44] C. Manksi. The structure of random utility models. *Theory and Decisions*, 8:229–254, 1977.

[45] J. Falmagne. A representation theorem for finite random scale systems. *Journal of Mathematical Psychology*, 18:52–72, 1978.

[46] S. Barbera and P. Pattanaik. Falmagne and the rationalizability of stochastic choices in terms of random orderings. *Econometrica*, 54:707–715, 1986.

[47] S. Mahajan and G.J. van Ryzin. Stocking retail assortments under dynamic consumer substitution. *Operations Research*, 49:334–351, 2001.

[48] J. Smith, C. Lim, and A. Alptekinoğlu. New product introduction against a predator: A bilevel mixed-integer programming approach. *Naval Research Logistics*, 56(8):714–729, 2009.

[49] D. Honhon, V. Gaur, and S. Seshadri. Assortment planning and inventory decisions under stockout-based substitution. *Operations Research*, 58(5):1364–1379, 2010.

[50] D. Honhon, S. Jonnalagedda, and X. Pan. Optimal algorithms for assortment selection under ranking-based consumer choice models. *M&SOM*, 14(2):279–289, 2012.

[51] S. Jagabathula and P. Rusmevichientong. A nonparametric joint assortment and price choice model. *Management Science*, 63(9):3128—-3145, 2017.

[52] V. Farias, S. Jagabathula, and D. Shah. Assortment optimization under general choice. SSRN Electronic Journal, February 2016.

[53] D. Bertsimas and V. Mišić. Exact first-choice product line optimization. *Operations Research*, 67(3):651–670, 2019.

[54] S.D. Jena, A. Lodi, H. Palmer, and Sole C. A partially-ranked choice model for large-scale data-driven assortment optimization. *INFORMS Journal on Optimization*, forthcoming, 2020.

[55] D. Zhang and W. Cooper. Revenue management for parallel flights with customer-choice behavior. *Operations Research*, 53(3):415–431, 2006.

[56] L. Chen and T. Homem de Mello. Mathematical programming models for revenue management under customer choice. *European Journal of Operational Research*, 203: 294–305, 2010.

[57] J. Chaneton and G. Vulcano. Computing bid-prices for revenue management under customer choice behavior. *M&SOM*, 13(4):452–470, 2011.

[58] S. Kunnumkal. Randomization approaches for network revenue management with customer choice behavior. *Production and Operations Management*, 23:1617–1633, 2014.

[59] V. Farias, S. Jagabathula, and D. Shah. A non-parametric approach to modeling choice with limited data. *Management Science*, 59(2):305–322, 2013.

[60] G. van Ryzin and G. Vulcano. A market discovery algorithm to estimate a general class of non-parametric choice model. *Management Science*, 61(2):281–300, 2015.

[61] G. van Ryzin and G. Vulcano. An expectation-maximization method to estimate a rank-based choice model of demand. *Operations Research*, 65(2):396–407, 2017.

[62] I. Sher, J. Fox, K. Kim, and P. Bajari. Partial identification of heterogeneity in preference orderings over discrete choices. Working paper 17346, National Bureau of Economic Research, Cambridge, MA, 2011.

[63] U. Genschel and W. Meeker. A comparison of maximum likelihood and median-rank regression for weibull estimation. *Quality Engineering*, 22(4):236–255, 2010.

[64] R. Anupindi, M. Dada, and S. Gupta. Estimation of consumer demand with stock-out based substitution: An application to vending machine products. *Marketing Science*, 17:406–423, 1998.

[65] C. Conlon and J. Mortimer. Demand estimation under incomplete product availability. *American Economic Journal-Microeconomics*, 5:1–30, 2013.

[66] C. Stefanescu. Multivariate customer demand: Modeling and estimation from censored sales. Working paper, European School of Management and Technology, Berlin, Germany, 2009.

[67] S. Jagabathula and P. Rusmevichientong. A nonparametric joint assortment and price choice model. *Management Science*, 63(9):3128–3145, 2017.

[68] A. Dempster, N. Laird, and D. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society B*, 39:1–38, 1977.

[69] W. Greene. *Econometric Analysis*. Prentice Hall, fifth edition, 2003.

[70] Xilei Zhao, Xiang Yan, Alan Yu, and Pascal Van Hentenryck. Prediction and behavioral analysis of travel mode choice: A comparison of machine learning and logit models. *Travel Behaviour and Society*, 20:22–35, 2020.

[71] John Cheesman and John Daniels. *UML components*. Addison-Wesley Reading, 2001.

[72] Lorenz T Biegler and Victor M Zavala. Large-scale nonlinear programming using ipopt: An integrating framework for enterprise-wide dynamic optimization. *Computers & Chemical Engineering*, 33(3):575–582, 2009.

[73] Pauli Virtanen, Ralf Gommers, Travis E Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, et al. Scipy 1.0: fundamental algorithms for scientific computing in python. *Nature methods*, 17(3):261–272, 2020.

[74] B. Meade, L. Lafayette, G. Sauter, and D. Tosello. Spartan hpc-cloud hybrid: Delivering performance and flexibility. University of Melbourne, Australia. Online resource https://doi.org/10.4225/49/58ead90dceaaa, 2017.

[75] T. Bodea, M. Ferguson, and L. Garrow. Choice-based revenue management: Data from a major hotel chain. *M&SOM*, 11(2):356–361, 2009.

[76] T. Kamishima. Nantonac collaborative filtering: recommendation based on order responses. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 583–588. ACM, 2003.

[77] O. Besbes, R. Phillips, and A. Zeevi. Testing the validity of a demand model: An operations perspective. *M&SOM*, 12(1):162–183, 2010.

[78] Wang Chi Cheung and David Simchi-Levi. Thompson sampling for online personalized assortment optimization problems with multinomial logit choice models. *Available at SSRN 3075658*, 2017.

[79] S. Jagabathula and G. Vulcano. A partial-order-based model to estimate individual preferences using panel data. *Management Science*, 64(4):1609–1628, 2018.

[80] F. Chierichetti, R. Kumar, and A. Tomkins. Discrete choice, permutations, and reconstruction. *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, 2018.

[81] Z. M. Shen P. Rusmevichientong and D. B. Shmoys. Dynamic assortment optimization with a multinomial logit choice model and capacity constraint. *Operations research*, pages 1666–1680, 2010.

[82] S. Jagabathula and P. Rusmevichientong. A nonparametric joint assortment and price choice model. *Management Science*, 63(9):3128–3145, 2017.

[83] James Dong, A Serdar Simsek, and Huseyin Topaloglu. Pricing problems under the markov chain choice model. *Production and Operations Management*, 28(1):157–175, 2019.