Dynamic Data-Driven Estimation of Non-Parametric Choice Models

Nam Ho-Nguyen[1] and Fatma Kılınç-Karzan[1]

[1]Tepper School of Business, Carnegie Mellon University, Pittsburgh, PA, 15213, USA.

February 13, 2017

Abstract

We study non-parametric models to estimate consumer choice behavior from observational data. These models have recently been introduced to overcome issues of suboptimal fit inherent in the traditional parametric models. Due to their minimal assumptions on consumer behavior, non-parametric models are shown to have improved predictive performance and thus draw growing interest in practice and academia. However, the generic nature of these models presents new computational challenges: learning an appropriate non-parametric model requires solving an optimization problem with a factorial number of variables. This is intractable even for small-scale problems with only a few items to sell. In this paper, we present a generic yet simple framework based on convex conjugacy, saddle point duality and online convex optimization to efficiently learn a non-parametric choice model from consumer choice data. Our method enjoys provable convergence guarantees (in terms of the number of iterations required) and extends naturally to the dynamic case where new observations are added to the data set. Nevertheless, each iteration of our method, as well as existing approaches from the literature, require solving a combinatorial subproblem. In order to provide a completely efficient method, we examine this combinatorial subproblem in detail and identify conditions on the assortment structure under which it can be solved efficiently.

1 Introduction

Learning a consumer choice model to explain and predict how customers choose between products is a key challenge in many business applications. Consumer choice models capture customers' complex substitution behavior; they are used to predict demand when customers buy substitutable products and face situations where their most preferred item is unavailable. The ability to capture substitution behavior has been demonstrated to increase average revenue [24, 30].

Choice models are critical components of assortment optimization in revenue management. Assortment optimization is the problem of selecting products to display to a customer, with the goal of maximizing revenue or purchase rates. It naturally arises in both online and offline retail sectors, as well as in online advertising for the selection of ads to display. Consequently, the problem of choosing the best assortment *given* a choice model has been the focus of much of the literature on assortment optimization, see e.g., [17, 27, 28].

Because of its critical role in assortment optimization, choice model estimation itself is equally important. Choice models specify the probability distribution of rankings that customers may have

for the products. These models often face an undesirable trade off between predictive accuracy and tractability: simple, but possibly inaccurate, choice models lead to easy optimization problems, while more complex choice models lead to intractability for estimation as well as the assortment optimization problem. Traditional estimation models follow a *parametric* approach where one first specifies a parametric structure for the probability distribution and then estimates the necessary parameters from past data. For example, the multinomial logit (MNL) model [4] is a classic and widely used parametric model, and offers an easy framework for choice model estimation. On the other hand, the MNL model assumes some unreasonable consumer behavior patterns, such as the independence of irrelevant alternatives property. As a remedy, parametric models with more complicated parametric structure, such as the nested logit model [19, 31] or the mixed MNL model [20], have been suggested and studied in the literature. Nevertheless, such more complicated parametric models come with it under/overfitting issues.

Due to these concerns on parametric models, the non-parametric approach, i.e., estimation of the probability distribution for rankings without imposing parametric structure, is attracting more attention [10, 25]. However, specifying a full non-parametric model is intractable for even moderate-sized n. To bypass this, we estimate a sparse non-parametric choice model that is close to the best fitting one; this our main focus in this paper. We next give a detailed overview of non-parametric choice models.

Non-Parametric Choice Models

In the general non-parametric choice estimation framework, we consider a firm which has n products to sell. For a positive integer $n \in \mathbb{N}$, we let $[n] := \{1, \dots, n\}$ and define S_n to be the collection of rankings of the set $[n]$. We assume that the firm chooses from a given set of m assortments $\mathcal{A}_1, \dots, \mathcal{A}_m \subset [n]$, and the item 1 represents the 'no-buy' option and is present in all assortments $\mathcal{A}_j$, $j \in [m]$. Therefore, when presented with an assortment, the customer will always choose an item from it (perhaps the no-buy option, i.e., item 1). We also denote $N := \sum_{j=1}^{m} |\mathcal{A}_j|$.

An incoming customer will choose an item according to his/her ranking $\sigma \in S_n$ of the products in $[n]$, that is, when presented with an assortment $\mathcal{A}_j$, the customer chooses the highest ranked product from $\mathcal{A}_j$, i.e., they choose $\arg\min_{i \in \mathcal{A}_j} \sigma(i)$. The key assumption in this model is that the ranking σ of each incoming customer is distributed i.i.d. according to some distribution λ on the set of all rankings S_n. Then λ represents the vector of probabilities of each $\sigma \in S_n$ being drawn, i.e., the probability that a customer will have a ranking σ is $\lambda(\sigma)$.

Non-parametric choice model immediately capture many existing consumer choice models, e.g., the MNL, mixed MNL, and nested logit models. Nevertheless, such a non-parametric estimation approach poses a major computational challenge. In particular, the number of different possible rankings of n products is $n!$, and hence λ is a vector in $\mathbb{R}^{n!}$. Thus, there are $n!$ unknowns to learn, a prohibitively large number even for moderate values of n.

Despite this computational challenge, the non-parametric approach is gaining more attention because it avoids the issues related to model selection and underfitting/overfitting and can be shown through case studies that it leads to substantial improvement in sales prediction accuracy (and thus revenue) over traditional parametric models [10].

Literature

Earliest studies on non-parametric choice models appear in the economics and psychology literatures, e.g., Block and Marschak [6]. They were introduced into operations literature by Mahajan and van Ryzin [18] who also showed that non-parametric models capture a number of parametric

models as special cases. A number of studies focus on the use of non-parametric models in assortment optimization . For example, Honhon et al. [13] present some polynomial-time algorithms for revenue maximization using the non-parametric choice model, under structural assumptions on the set of possible preference profiles. However, Aouad et al. [3] recently show that, in general, the assortment optimization is NP-hard under the non-parametric choice model, *given* that we have a choice estimate.

There are a number different approaches employed for the *static* estimation of non-parametric choice models. Farias et al. [10] outline a method to estimate the non-parametric model from realistic consumer data based on constraint sampling. Other methods based on maximum likelihood estimation and norm minimization are presented in van Ryzin and Vulcano [29] and Bertsimas and Mišic [5] respectively. While Farias et al. [10] and van Ryzin and Vulcano [29] provide useful recovery results for the non-parametric model, none of the methods proposed in Farias et al. [10], van Ryzin and Vulcano [29] or Bertsimas and Mišic [5] come with provable convergence guarantees. Moreover, these methods are not yet equipped to deal with the dynamic data setting, that is, the case where the firm continuously collect and wishes to utilize more consumer data even as we estimate the non-parametric choice model.

In all of the literature on estimation of non-parametric choice models, one of the major challenges is to repeatedly solve an NP-hard combinatorial optimization subproblem with a factorial number of unknowns in the number of products. Even for relatively small-scale models, this can be prohibitively large. Méndez-Díaz et al. [21] recently analyzed the polyhedral structure of this combinatorial subproblem, and based on this suggested a branch-and-cut algorithm. Nevertheless, no polynomial-time solvable cases of this problem was known before.

Our Contributions

In this paper, we outline a method to dynamically estimate non-parametric choice models which addresses some of the concerns on the approaches of Farias et al. [10], van Ryzin and Vulcano [29] and Bertsimas and Mišic [5] raised above. Our contributions are as follows.

1. We present a unified view of the methods from Farias et al. [10], van Ryzin and Vulcano [29] and Bertsimas and Mišic [5]. In particular, we show that the same NP-hard combinatorial subproblem arises in all three methods.

2. We propose a general framework for estimating non-parametric choice models via a general distance function. We also propose a solution method based on saddle point duality, convex conjugacy, online convex optimization. Our method enjoys provably efficient convergence guarantees, which also upper-bounds the sparsity of our estimated model. As a result, our analysis exposes an explicit trade-off between the sparsity of the non-parametric choice model λ and its estimation accuracy ϵ for the first time.

3. Furthermore, by employing the joint estimation-optimization technique, we show that our method can adapt easily and efficiently to the dynamic data setting, where our data are updated continuously.

4. Our method encounters the same NP-hard combinatorial subproblem that the three previous methods also encounter. Therefore, we examine this combinatorial subproblem in detail. As opposed to the polyhedral study of Méndez-Díaz et al. [21], in order to identify the cases where a completely efficient algorithm can be utilized, we identify various structural conditions on

assortments under which the combinatorial subproblem can be solved in polynomial-time and we present the corresponding polynomial-time solution procedures. These cases include, for example, various tree-like structures on the underlying line graph of the assortments.

Outline

In Section 2, we introduce our notation and discuss our model and data collection process. In Section 3, we outline the methods of Farias et al. [10], van Ryzin and Vulcano [29] and Bertsimas and Mišic [5] to estimate non-parametric choice models and highlight the similarities and differences between these methods. Moreover, we point out how the combinatorial subproblem arises within all these methods. In Section 4, we outline our approach and establish the number of iterations needed to achieve a given solution accuracy ϵ. Along the way, we also point out how the combinatorial subproblem arises within our framework. We dedicate Section 5 to the combinatorial subproblem, where we explore the subproblem in detail and identify a number of conditions on the assortment structures which ensures existence of polynomial-time algorithms and present these algorithms. Finally, we summarize our results and outline some further directions in Section 6. We defer all proofs to Appendix A.

2 Notation and Preliminaries

2.1 Notation

For a positive integer $n \in \mathbb{N}$, we let $[n] = \{1, \ldots, n\}$, define $\Delta_n := \{x \in \mathbb{R}^n_+ : \sum_{i\in[n]} x_i = 1\}$ to be the standard simplex, and S_n to be the collection of rankings of the set $[n]$. We refer to a collection of objects b_j, $j \in J$ by the notation $\{b_j\}_{j\in J}$. Throughout the paper, the superscript, e.g., y^t, z^t, f^t, is used to attribute items to the t-th time period or iteration. The subscript is used to denote coordinates of a vector or matrix, e.g., a_{ij}. Given vectors x and y, $\langle x, y\rangle$ corresponds to the usual inner product of x and y. Given a norm $\|\cdot\|$ on a Euclidean space $\mathbb{E}$ and a real number $r > 0$, we represent the $\|\cdot\|$-norm ball of radius r by $B_{\|\cdot\|}(r) = \{x \in \mathbb{E} : \|x\| \le r\}$, and denote its dual norm by $\|x\|_* = \min_{y\in B_{\|\cdot\|}(1)}\langle x, y\rangle$. We let $\partial f(x)$ be the subdifferential of f taken at x. We abuse notation slightly by denoting $\nabla f(x)$ for both the gradient of function f at x if f is differentiable and a subgradient of f at x, even if f is not differentiable. If ϕ is of the form $\phi(x, y)$, then $\nabla_x\phi(x, y)$ denotes the subgradient of ϕ at x while keeping the other variables fixed at y. We denote the indicator function as $\mathbb{I}$, i.e., $\mathbb{I}(\mathcal{S}) = 1$ if statement $\mathcal{S}$ holds, and $\mathbb{I}(\mathcal{S}) = 0$ otherwise.

2.2 Data Collection Process

Let us define some important notation relating the vector of probabilities λ to items i and assortments $\mathcal{A}_j$. Given a ranking σ, we denote $i_j(\sigma) = \arg\min_{i\in\mathcal{A}_j} \sigma(i)$ to be the top ranked item in $\mathcal{A}_j$ with respect to the ranking σ. For an item-assortment pair $i \in \mathcal{A}_j$, we define a 0-1 vector $a_{ij} \in \mathbb{R}^{n!}$ to have entries $a_{ij}(\sigma) = \mathbb{I}(i = i_j(\sigma))$ for $\sigma \in S_n$. Thus, each entry of a_{ij} corresponds to a ranking σ, with $a_{ij}(\sigma) = 1$ if i is the highest ranked item in $\mathcal{A}_j$ according to σ, and $a_{ij}(\sigma) = 0$ otherwise. We define A to be the 0-1 matrix of dimension $N \times n!$ with rows $a_{ij}^\top$. Given a ranking σ, we define the vector $a(\sigma) \in \{0, 1\}^N$ by the same formula: $a_{ij}(\sigma) = \mathbb{I}(i = i_j(\sigma))$, thus $a(\sigma)$ are simply the columns of the matrix A. Finally, for $j \in [m]$, we define A_j to be the submatrix of A with rows $a_{ij}^\top$ for $i \in \mathcal{A}_j$.

Then, based on our notation, given a distribution $\lambda \in \Delta_{n!}$ over rankings, the probability $\mathbb{P}_\lambda[i \mid \mathcal{A}_j]$ that a customer chooses item i from $\mathcal{A}_j$ is simply represented as the inner product $\langle a_{ij}, \lambda\rangle$. Moreover, $A\lambda$ is nothing but the vector composed of the collection of probabilities

$\{\mathbb{P}_\lambda[i \mid \mathcal{A}_j]\}_{i\in\mathcal{A}_j, j\in[m]}$. Finally, the probability distribution of a customer choosing item $i \in \mathcal{A}_j$ given that they were offered assortment $\mathcal{A}_j$, i.e. $\{\mathbb{P}_\lambda[i \mid \mathcal{A}_j]\}_{i\in\mathcal{A}_j}$, is simply $A_j\lambda$.

Our environment may necessitate the form of the data collection process which can lead to a number of different types of data to learn the model from (see [10, Sections 2.2, 6.2]). In this paper, we assume the following on the data collection process. When a customer arrives, the firm displays an assortment $\mathcal{A}_j$ to the customer. The customer chooses a product $i_j \in \mathcal{A}_j$, and the firm observes this choice.

A data set with K such observations will be a collection of pairs $\left\{i^k, \mathcal{A}^k\right\}_{k=1}^K$, where i^k denotes the item chosen and $\mathcal{A}^k$ denotes the assortment displayed for observation k. There are a number of useful statistics on this data set, which are defined as follows:

$$q_{ij} := \frac{1}{K}\sum_{k=1}^{K} \mathbb{I}(i^k = i, \mathcal{A}^k = \mathcal{A}_j) \tag{1}$$

$$q_j := \frac{1}{K}\sum_{k=1}^{K} \mathbb{I}(\mathcal{A}^k = \mathcal{A}_j) \tag{2}$$

$$p_{ij} := \frac{\sum_{k=1}^{K} \mathbb{I}(i^k = i, \mathcal{A}^k = \mathcal{A}_j)}{\sum_{k=1}^{K} \mathbb{I}(\mathcal{A}^k = \mathcal{A}_j)} = \frac{q_{ij}}{q_j}. \tag{3}$$

In words, q_{ij} is the proportion of observations where assortment $\mathcal{A}_j$ was displayed and item i was chosen, q_j is the proportion of observations where assortment $\mathcal{A}_j$ was displayed, and p_{ij} is the proportion of customers who chose item i *given that* assortment $\mathcal{A}_j$ was displayed. These statistics will be used to infer the best-fitting probability distribution λ. We denote the collected vectors of $\{p_{ij}\}_{i\in\mathcal{A}_j, j\in[m]} = p \in \mathbb{R}^N$ and $\{p_{ij}\}_{i\in\mathcal{A}_j} = p_j \in \mathbb{R}^{|\mathcal{A}_j|}$ for $j \in [m]$.

3 Existing Approaches to Non-parametric Choice Estimation

In this section, we examine three different approaches that are closest to our work from the existing literature to learn the non-parametric choice model, i.e., infer an appropriate probability vector λ using the data collected via the process outlined in Section 2.2.

3.1 Revenue Prediction Approach

Let r_i be the revenue of item $i \in [n]$. Then the expected revenue of assortment $\mathcal{A}_j$ under distribution λ is

$$\sum_{i\in\mathcal{A}_j} r_i \mathbb{P}_\lambda[i \mid \mathcal{A}_j] = \sum_{i\in\mathcal{A}_j} r_i \langle a_{ij}, \lambda\rangle.$$

Farias et al. [10] seek to find the worst-case expected revenue from a distribution λ consistent with the given data in the sense that the theoretical probabilities $\mathbb{P}_\lambda[i \mid \mathcal{A}_j]$ are consistent with their empirical estimates p_{ij}. Thus, they wish to solve the linear program (LP)

$$\min_\lambda \left\{ \sum_{i\in\mathcal{A}_j} r_i \langle a_{ij}, \lambda\rangle : \; A\lambda = p, \; \lambda \in \Delta_{n!} \right\}. \tag{4}$$

The optimization problem (4) is computationally intractable even for moderate values of n because it involves $n!$ variables. Nonetheless, the dual of (4) admits the following robust LP interpretation:

$$\max_{\beta,\nu} \left\{ \langle \beta, p \rangle - \nu + \sum_{i \in \mathcal{A}_j} r_i p_{ij} : \max_{\sigma \in S_n} \langle \beta, a(\sigma) \rangle \leq \nu \right\}. \tag{5}$$

Observation 3.1. Note that verifying the feasibility of a solution with respect to the robust constraint in (5), i.e.,

$$\max_{\sigma \in S_n} \langle \beta, a(\sigma) \rangle = \max_{\sigma} \left\{ \sum_{j \in [m]} \sum_{i \in \mathcal{A}_j} \beta_{ij} a_{ij}(\sigma) : \sigma \in S_n \right\} \leq \nu, \tag{RM-RC(β, ν))}$$

requires solving a combinatorial subproblem. In our context, this subproblem occurs frequently. In Section 5, we will show that (RM-RC(β, ν)) is NP-hard and we will study its properties further. ∎

Farias et al. [10] suggests solving (5) using the constraint sampling technique from [7] or by building an approximation to its robust counterpart obtained from approximating the uncertainty sets with an efficiently representable polyhedron.

The revenue prediction approach of Farias et al. [10] is shown to exhibit unique recovery guarantees under a particular *signature condition* on the true probability vector λ. The probability vector λ is said to satisfy the signature condition if, for each ranking $\sigma \in S_n$ with positive probability $\lambda(\sigma) > 0$, there exists an item-subset pair $i \in \mathcal{A}_j$ such that i is top-ranked in $\mathcal{A}_j$ by σ (i.e., $i = i_j(\sigma)$), but for any other $\sigma' \in S_n$ with positive probability $\lambda(\sigma') > 0$, i is not top ranked in $\mathcal{A}_j$ by σ'. Furthermore, under this signature condition and when the assortments $\mathcal{A}_j$ consists of all pairs of items (i.e., comparison data), an efficient algorithm to find such λ is described in Farias et al. [9]. However, the signature condition is hard to justify in general, and there is no guarantee that there exists $\lambda \in \Delta_{n!}$ to fit the data p exactly, i.e., $A\lambda = p$. To remedy this, van Ryzin and Vulcano [29] and Bertsimas and Mišic [5] present alternative approaches to learn λ without relying on the signature condition or enforcing the constraint $A\lambda = p$ precisely.

3.2 Maximum Likelihood Estimation Approach

van Ryzin and Vulcano [29] propose the following method to learn λ via maximum likelihood estimation (MLE). We next describe their method and provide an alternative interpretation of their approach as the minimization of a particular distance measures, namely Kullback-Leibler (KL) divergence, between the true distributions $A_j\lambda$ and their empirical estimates p_j.

Recall that for each observation k, we have an item-assortment pair $(i^k, \mathcal{A}^k)$, which corresponds to some item subset pair $(i^k, \mathcal{A}^k) = (i, \mathcal{A}_j)$ in our usual indices $i \in [n]$, $j \in [m]$. Then for some distribution λ on rankings, the likelihood of observation $(i^k, \mathcal{A}^k)$ occurring, given that $\mathcal{A}^k$ is displayed, is $\mathbb{P}_\lambda[i^k \mid \mathcal{A}^k] = \mathbb{P}_\lambda[i \mid \mathcal{A}_j] = \langle a_{ij}, \lambda \rangle$. Conversely, each pair $(i, \mathcal{A}_j)$ occurs some number of times amongst the observations $\{i^k, \mathcal{A}^k\}_{k=1}^K$; in particular, by (1), this number is Kq_{ij}. Now, the likelihood of observing the dataset $\{i^k, \mathcal{A}^k\}_{k=1}^K$ is the product of the terms $\mathbb{P}_\lambda[i^k \mid \mathcal{A}^k]$, and hence its log-likelihood is the sum of the log-probabilities. Based on our observations, we can write the log-likelihood in terms of our original set of indices $\sum_{j \in [m]} \sum_{i \in \mathcal{A}_j} Kq_{ij} \log(\langle a_{ij}, \lambda \rangle)$. Thus, ignoring

the constant K factor, the MLE problem is

$$\max_{\lambda} \left\{ \sum_{j\in[m]} \sum_{i\in\mathcal{A}_j} q_{ij} \log\left(\langle a_{ij}, \lambda\rangle\right) : \ \lambda \in \Delta_{n!} \right\}. \tag{6}$$

Throughout, we use the convention that $x\log(x) = 0$ when $x = 0$ to avoid ambiguity when we have $q_{ij} = \langle a_{ij}, \lambda\rangle = 0$, in such a case we set $q_{ij}\log(\langle a_{ij}, \lambda\rangle) = 0$. This implies that if the optimal solution λ to (6) has $\mathbb{P}_\lambda[i \mid \mathcal{A}_j] = \langle a_{ij}, \lambda\rangle = 0$, then we must have $q_{ij} = 0$ als, i.e., we did not observe any choices of i from $\mathcal{A}_j$ in our data either.

Like (4), the problem (6) is very large, with $n!$ variables. A column generation technique is suggested by van Ryzin and Vulcano [29] to get around this. They start by solving (6) on a subset of the variables $\lambda(\sigma)$, $\sigma \in S \subset S_n$, to get a solution $\lambda(S)$. Then, from the optimality conditions for (6), the MLE column generating (MLE-CG) subproblem is constructed as

$$\max_{\sigma} \left\{ \sum_{j\in[m]} \sum_{i\in\mathcal{A}_j} \frac{q_{ij} a_{ij}(\sigma)}{\langle a_{ij}, \lambda(S)\rangle} : \ \sigma \in S_n \right\}. \tag{MLE-CG(S)}$$

The solution $\lambda(S)$ is optimal if (MLE-CG(S)) $\le K$, otherwise the column σ^* maximizing (MLE-CG(S)) is added to the set S, and the process is repeated. Note that the column generating subproblem (MLE-CG(S)) is the same NP-hard combinatorial subproblem as (RM-RC(β, ν)).

Observation 3.2. The MLE problem (6) admits a nice interpretation between the empirical estimates $\{p_j\}_{j\in[m]}$ and the distributions $\{A_j\lambda\}_{j\in[m]}$. To observe this, let us rewrite (6) as

$$\begin{aligned}
&\max_{\lambda} \left\{ \sum_{j\in[m]} \sum_{i\in\mathcal{A}_j} q_{ij} \log\left(\langle a_{ij}, \lambda\rangle\right) : \ \lambda \in \Delta_{n!} \right\} \\
&= \max_{\lambda} \left\{ \sum_{j\in[m]} q_j \sum_{i\in\mathcal{A}_j} p_{ij} \log\left(\langle a_{ij}, \lambda\rangle\right) : \ \lambda \in \Delta_{n!} \right\} \\
&= \max_{\lambda} \left\{ -\sum_{j\in[m]} q_j \sum_{i\in\mathcal{A}_j} p_{ij} \log\left(\frac{p_{ij}}{\langle a_{ij}, \lambda\rangle}\right) : \ \lambda \in \Delta_{n!} \right\} + \sum_{j\in[m]} q_j \sum_{i\in\mathcal{A}_j} p_{ij}\log(p_{ij}) \\
&= -\min_{\lambda} \left\{ \sum_{j\in[m]} q_j \,\mathrm{KL}(p_j, A_j\lambda) : \ \lambda \in \Delta_{n!} \right\} + \text{const},
\end{aligned}$$

where $\mathrm{KL}(a, b)$ is the Kullback-Leibler (KL) divergence between two probability distributions a and b, and const $= \sum_{j\in[m]} q_j \sum_{i\in\mathcal{A}_j} p_{ij}\log(p_{ij})$. Hence, (6) is equivalent to solving

$$\min_{\lambda} \left\{ \sum_{j\in[m]} q_j \,\mathrm{KL}(p_j, A_j\lambda) : \ \lambda \in \Delta_{n!} \right\}. \tag{7}$$

Recall that $A_j\lambda$ is the probability distribution of a customer selecting item $i \in \mathcal{A}_j$ when assortment $\mathcal{A}_j$ is presented, and p_j is their distribution's empirical estimates. This minimization problem can be interpreted as a weighted sum of the 'distance measures' (i.e., the KL divergences) between the true distributions $A_j\lambda$ and their empirical estimates p_j. ∎

3.3 Norm-Minimization Approach

As opposed to the approaches outline in Sections 3.1 and 3.2, in order to estimate a non-parametric choice model λ, Bertsimas and Mišic [5] suggest minimizing the ℓ_1-norm of $p - A\lambda$ by solving

$$\min_{\lambda} \{\|p - A\lambda\|_1 : \ \lambda \in \Delta_{n!}\}. \tag{8}$$

Observation 3.3. The approach of Bertsimas and Mišic [5] can be interpreted in the same way as (7) in Observation 3.2, but replacing the KL divergence distance measure and the weights q_j with different distance measures and weights. To recover (8) from (7), the distance measures are chosen as the ℓ_1-norm and the weights are set to be uniform. Notice that then we no longer write a sum over $j \in [m]$ since $\sum_{j\in[m]} \|p_j - A_j\lambda\|_1 = \|p - A\lambda\|_1$. ∎

In fact, (8) can be cast as a LP, but it is still computationally intractable since the dimension of λ is $n!$. Similar to van Ryzin and Vulcano [29], Bertsimas and Mišic [5] address this computational difficulty via a column generation approach. Their column generating subproblem is of the form

$$\max_{\sigma} \left\{ \sum_{j\in[m]} \sum_{i\in\mathcal{A}_j} \beta_{ij}(S) a_{ij}(\sigma) - \nu(S) : \ \sigma \in S_n \right\}, \tag{NM-CG(S))}$$

where $\beta(S), \nu(S)$ are from the dual solution to solving (8) on a subset of columns $\sigma \in S \subset S_n$. If the objective of (NM-CG(S)) is > 0, then the column σ^* corresponding to the optimal solution of (NM-CG(S)) is added to the set S and the process is repeated. Once again, note that the column generating subproblem (NM-CG(S)) is an instance of the same class of NP-hard combinatorial subproblems that (RM-RC(β, ν)) and (MLE-CG(S)) belong to.

3.4 Discussion of the Existing Approaches

The fact that the same combinatorial subproblem arises in all of these approaches in Sections 3.1–3.3 is quite remarkable. This highlights that there may be a fundamental connection behind all these approaches and suggests that possibly all of them can be viewed and treated in a unified manner.

Farias et al. [10] and van Ryzin and Vulcano [29] provide *recovery* guarantees for their approaches. However, the three methods outlined above lack formal convergence guarantees. Furthermore, they are not flexible enough to immediately adapt to changes in the data set: for example when new observations are added, one has to repeat their entire solution process on the new data set. In particular, they cannot utilize their choice model estimated with the previous data in a meaningful and efficient manner. While Farias et al. [10] mention a robust version of revenue estimation to handle data uncertainty, the problem of an evolving data set, which is of great practical interest, is not addressed.

In the next section, we outline an approach for non-parametric choice model estimation which not only brings together the seemingly disparate existing approaches but also addresses the two shortcomings outlined above of all of the existing approaches.

4 Dynamic Learning of a Non-Parametric Choice Model

Our more general approach to learning a non-parametric choice model is based on

$$\min_{\lambda} \{D(A\lambda, p) : \ \lambda \in \Delta_{n!}\}, \tag{9}$$

where $D(\cdot,\cdot)$ is a general distance measure between two points. We assume that $D(\cdot,p)$ is convex and continuous on its domain.

This approach immediately generalizes the approaches in Sections 3.2 and 3.3 because we can define $D(A\lambda,p)$ to be $\sum_{j\in[m]} q_j \,\mathrm{KL}(p_j, A_j\lambda)$ and $\|p - A\lambda\|_1$ respectively to arrive at (7) and (8).

For a specified accuracy level $\epsilon > 0$, our goal is to obtain an (additive error) ϵ-approximate solution to this problem within a reasonable number of iterations. We also consider a dynamic variant of the problem where instead of fixed data p, we now have changing data p^t that converges to a limit p as our observations increase, i.e., $t \to \infty$. In this setup we would still like to estimate using p, but we are only given access to the sequence $\{p^t\}_{t\geq 1}$. To address these two considerations, we use three tools: convex conjugacy, online convex optimization, and joint estimation-optimization.

4.1 Saddle Point Formulation via Convex Conjugacy

Given that $D(\cdot,p)$ is convex and continuous on its domain, we can write $D(A\lambda,p)$ via its convex conjugate:

$$D(A\lambda,p) = \sup_y \left\{\langle A\lambda, y\rangle - D^*(y,p) :\ y\in\mathbb{R}^N\right\},$$
$$\text{where}\quad D^*(y,p) := \sup_z \left\{\langle y, z\rangle - D(z,p) :\ z\in\mathbb{R}^N\right\}.$$

Note that $D^*(y,p)$ is convex in y. In addition, if we assume that the (sub)gradients $\nabla_z D(z,p)$ are bounded $\|\nabla_z D(z,p)\| \leq R$ for some norm $\|\cdot\|$, then we can restrict the domain of y accordingly:

$$D(A\lambda,p) = \sup_{y:\|y\|\leq R} \left\{\langle A\lambda, y\rangle - D^*(y,p) :\ y\in\mathbb{R}^N\right\}.$$

Henceforth, we denote $Y := \{y :\ \|y\| \leq R\}$ to be the corresponding norm ball. Based on these definitions, the problem (9) now admits a natural saddle point representation:

$$\mathrm{SV}(p) := \min_\lambda \{D(A\lambda,p) :\ \lambda\in\Delta_{n!}\} = \min_{\lambda\in\Delta_{n!}} \sup_{y\in Y} \{\langle A\lambda, y\rangle - D^*(y,p)\}. \tag{10}$$

In an ideal situation, we would solve (10) with a saddle point algorithm such as Mirror Prox [23], which leads to a convergence rate of $O\left(\log(m+n!)/T\right) \approx O\left((\log(m) + n\log(n))/T\right)$ after T iterations. However, each iteration of the Mirror Prox algorithm involves updating λ, which is very expensive, taking at least $O(n!)$ time. This is intractable even for moderate values of n. In addition, due to the nature of the operations involved in Mirror Prox algorithm, the resulting solution for λ will always be dense, introducing storage issues for vectors in $\mathbb{R}^{n!}$. Randomization approaches developed for Mirror Prox [16] could offer a partial remedy for this. Yet, most randomized algorithms require us to sample over a vector in $\mathbb{R}^{n!}$, which again takes $O(n!)$ time.

To address this, we first transform the problem using standard convex conjugacy. The minimax theorem [26] allows us to write (10) as

$$\mathrm{SV}(p) = \sup_{y\in Y} \min_{\lambda\in\Delta_{n!}} \{\langle A\lambda, y\rangle - D^*(y,p)\} = \sup_{y\in Y} \min_{\sigma\in S_n} \{\langle a(\sigma), y\rangle - D^*(y,p)\}.$$

The second equality follows from the fact that $\langle A\lambda, y\rangle - D^*(y,p)$ is a linear function in λ, thus the minimum occurs at a vertex of $\Delta_{n!}$, i.e., some column of A. Thus, we arrive at

$$\mathrm{SV}(p) = \sup_{y\in Y} f(y,p), \tag{11}$$

where

$$f(y,p) := \min_{\sigma \in S_n} \{\langle a(\sigma), y\rangle - D^*(y,p)\}.$$

Notice that $f(y,p)$ is a concave function in y with supergradients $\nabla_y f(y,p) = a(\sigma) - \nabla_y D^*(y,p)$, where $\sigma \in \arg\min_{\sigma' \in S_n} \langle a(\sigma'), y\rangle$. Thus, $\mathrm{SV}(p)$ is simply maximizing a concave function over a bounded convex domain Y.

By exploiting a link between convex conjugacy and online convex optimization, we can relate the solution of (11) to the original problem (9). We note that similar ideas ideas were applied to online stochastic programming [1] and in an algorithmic approach to Approximate Caratheodory Theorem [22].

In the dynamic variant of (11), we do not assume we have access to a fixed data vector p, but instead we are given a sequence $\{p^t\}_{t\geq 1}$, which arises when we collect new data and update our empirical distributions. Under standard statistical assumptions, the estimates p^t obtained from a growing set of observations converge to a 'true' distribution vector p (almost surely). Thus, the dynamic variant can be described as follows: we would like to maximize $f(y,p)$ over Y, given that we only have access to approximate data $p^t \approx p$. A naïve approach would be to fix a p^t then maximize $f(y,p^t)$. However, this introduces an issue of consistency: no matter how many iterations we perform, our solution will never converge to $\sup_{y\in Y} f(y,p)$. To overcome this, we could re-optimize each time we receive a new p^t, but this is quite inefficient since we cannot exploit previous solutions y^τ obtained for maximizing $f(y,p^\tau)$ where $\tau \in [t-1]$. As opposed to this, the joint estimation-optimization (JEO) framework [2, 12] optimizes $f(y,p)$ using a sequence $p^t \to p$ without having to repeatedly solve similar problems each time we receive a new p^t.

Since online convex optimization (OCO) plays a crucial role in our analysis, we first give a brief introduction to OCO and first-order methods in Section 4.2. In Section 4.3, we bring these ideas together and present our solution approach for (9).

4.2 Online Convex Optimization

Online convex optimization (OCO) is commonly used to capture decision making in dynamic environments. Here we outline the basic OCO concepts; for further details and background, we refer the reader to Cesa-Bianchi and Lugosi [8].

In OCO, we are given a finite time horizon T, closed, bounded, and convex domain Z, and in each time period $t \in [T]$, a convex loss function $f^t : Z \to \mathbb{R}$ is revealed. At time periods $t \in [T]$ we must choose a decision $z^t \in Z$, and based on this we suffer a loss of $f^t(z^t)$ and receive some feedback typically in the form of first-order information on f^t. The main aim of OCO is to choose a sequence of points $\{z^t\}_{t=1}^T$ from the domain $\mathcal{Z}$ to bound the weighted regret

$$\sum_{t=1}^{T} \theta^t f^t(z^t) - \inf_{z\in\mathcal{Z}} \sum_{t=1}^{T} \theta^t f^t(z), \tag{12}$$

where $\theta = \{\theta^t\}_{t=1}^T \in \Delta_T$ is a collection of convex combination weights. The key restriction that separates OCO from standard optimization problems is that z^t must be chosen *before* observing f^t. The fact that there exists algorithms which bound (12) for *any* sequence $\{f^t\}_{t=1}^T$ is the crucial aspect of OCO which we exploit to solve the dynamic variant of (11).

A key class of algorithms which can be used for OCO (as well as standard offline convex optimization) are first-order methods (FOMs). Following the notation in the surveys [14, 15], we outline the proximal setup for a general domain $\mathcal{Z}$. This setup forms the basis for several FOMs such as Mirror Descent and is used in their convergence analyses.

ALGORITHM 1: Generalized Mirror Descent

Input: time horizon T, positive step sizes $\{\gamma^t\}_{t=1}^T$, and a sequence of vectors $\{\xi^t\}_{t=1}^T$
Output: sequence $\{z^t\}_{t=1}^T$ from $\mathcal{Z}$.
$z^1 := \min_{z\in\mathcal{Z}} \omega(z)$;
for $t = 1, \dots, T$ **do**
    $z^{t+1} = \mathrm{Prox}_{z^t}(\gamma^t \xi^t)$;
end

- *Norm*: $\|\cdot\|$ on the Euclidean space $\mathbb{E}$ where the domain $\mathcal{Z}$ lives, along with its dual norm $\|\zeta\|_* := \max\limits_{\|z\|\le 1} \langle \zeta, z\rangle$.
- *Distance-Generating Function* (d.g.f.): A function $\omega(z) : \mathcal{Z} \to \mathbb{R}$, which is convex and continuous on $\mathcal{Z}$, and admits a selection of subgradients $\nabla\omega(z)$ that is continuous on the set $\mathcal{Z}^\circ := \{z \in \mathcal{Z} : \partial\omega(z) \neq \emptyset\}$ (here $\partial\omega(z)$ is a subdifferential of ω taken at z), and is strongly convex with modulus 1 with respect to $\|\cdot\|$:
$$\forall z', z'' \in \mathcal{Z}^\circ : \ \langle \nabla\omega(z') - \nabla\omega(z''),\ z' - z''\rangle \ge \|z' - z''\|^2.$$
- *Prox-mapping*: Given a *prox center* $z \in \mathcal{Z}^\circ$,
$$\mathrm{Prox}_z(\xi) := \arg\min_{z'\in\mathcal{Z}} \left\{\langle \xi - \nabla\omega(z), z'\rangle + \omega(z')\right\} : \mathbb{E} \to \mathcal{Z}^\circ.$$
When the d.g.f. is taken as the squared ℓ_2-norm, the prox mapping becomes the usual projection operation of the vector $z - \xi$ onto $\mathcal{Z}$.
- *Set width*: $\Omega = \Omega_z := \max\limits_{z\in\mathcal{Z}} \omega(z) - \min\limits_{z\in\mathcal{Z}} \omega(z)$.

For common domains $\mathcal{Z}$ such as simplex, Euclidean ball, and spectahedron, standard proximal setups, i.e., selection of norm $\|\cdot\|$, d.g.f. $\omega(\cdot)$, the resulting Prox computations and set widths Ω are discussed in [14, Section 1.7].

In the most basic setup, our functions f^t are convex and non-smooth. In this case, we utilize a generalization of Mirror Descent, outlined in Algorithm 1 for bounding the weighted regret (12).

We next state a bound on the weighted regret (12) in the most general case where our functions f^t need only satisfy convexity and Lipschitz continuity. More precisely, we will assume the following.

Assumption 4.1. A proximal setup of Section 4.2 exists for the domain $\mathcal{Z}$. Each function f^t is convex, and there exists $G \in (0, \infty)$ such that the subgradients of f^t are bounded, i.e., $\|\nabla f^t(z)\|_* \le G$ for all $z \in \mathcal{Z}$ and $t \in [T]$.

Theorem 4.1 ([12, Theorem 1])**.** *Suppose Assumption 4.1 holds, and we are given weights $\theta \in \Delta_T$. Then running Algorithm 1 with $\xi^t = \theta^t \nabla f^t(z^t)$, and step sizes $\gamma_t = \gamma := \sqrt{\frac{2\Omega}{\sup_{t\in[T]}(\theta^t)^2 G^2 T}}$ for all $t \in [T]$ results in*
$$\sum_{t=1}^T \theta^t f^t(z^t) - \inf_{z\in\mathcal{Z}} \sum_{t=1}^T \theta^t f^t(z) \le \sqrt{2\Omega \left(\sup_{t\in[T]} (\theta^t)^2\right) G^2 T}.$$

The bound on weighted regret in Theorem 4.1 is optimized when the convex combination weights $\theta \in \Delta_T$ are set to be *uniform*, i.e., $\theta^t = 1/T$; in this case, the right hand side of the inequality above becomes $O(1/\sqrt{T})$.

4.3 Utilizing OCO for Dynamic Non-Parametric Choice Modeling

In Section 4.1, we established the equivalence between (9) and (11), where

$$f(y,p) = \min_{\sigma \in S_n} \{\langle a(\sigma), y\rangle - D^*(y,p)\}.$$

We set the sequence of functions f^t based on the current y^t as follows:

$$f^t(y) = \langle y, a(\sigma^t)\rangle - D^*(y,p^t), \quad \text{where} \quad \sigma^t = \arg\min_{\sigma \in S_n} \langle y^t, a(\sigma)\rangle. \tag{13}$$

If the proximal setup of Section 4.2 exists for Y and the (super)gradients are bounded $\|\nabla_y f^t(y)\|_* = \|a(\sigma^t) - \nabla_y D^*(y;p^t)\| \le G$ for all $y \in Y$ and data vectors p^t, then we obtain the following regret bound as a direct consequence of Theorem 4.1.

Lemma 4.2. *Suppose Assumption 4.1 holds for Y and f^t as defined in (13), and set uniform weights $\theta^t = 1/T$. Then running Algorithm 1 with $\xi^t = -\theta^t \nabla f^t(z^t)$, and $z^t = y^t$, step sizes $\gamma^t = \gamma := \sqrt{\frac{2\Omega}{\sup_{t\in[T]}(\theta^t)^2 G^2 T}}$ for all $t \in [T]$ results in*

$$\max_{y\in Y} \frac{1}{T}\sum_{t=1}^T f^t(y) - \frac{1}{T}\sum_{t=1}^T f^t(y^t) \le \sqrt{\frac{2\Omega G^2}{T}}.$$

The regret bound of Lemma 4.2 is instrumental in the derivation of the following result.

Theorem 4.3. *Let the sequence $\{y^t\}_{t=1}^T$ be generated according to Lemma 4.2. Then*

$$\begin{aligned} &D\left(\frac{1}{T}\sum_{t=1}^T a(\sigma^t), p\right) - \min_{\lambda\in\Delta_{n!}} D(A\lambda, p) \\ &\le \sqrt{\frac{2\Omega G^2}{T}} + \max_{y\in Y} \frac{1}{T}\sum_{t=1}^T \left[D^*(y,p^t) - D^*(y,p)\right] + \frac{1}{T}\sum_{t=1}^T \left[\min_{\lambda\in\Delta_{n!}} D(A\lambda, p^t) - \min_{\lambda\in\Delta_{n!}} D(A\lambda, p)\right]. \end{aligned}$$

Theorem 4.3 states that we can optimize the model for data p despite only given access to estimates p^t, and we will get better approximations as $T \to \infty$. In this convergence rate, as opposed to working with exact data p, the penalty for using inexact data p^t is captured in the two error terms

$$\max_{y\in Y} \frac{1}{T}\sum_{t=1}^T \left[D^*(y,p^t) - D^*(y,p)\right] \quad \text{and} \quad \frac{1}{T}\sum_{t=1}^T \left[\min_{\lambda\in\Delta_{n!}} D(A\lambda, p^t) - \min_{\lambda\in\Delta_{n!}} D(A\lambda, p)\right].$$

As $p^t \to p$, these two terms converge to zero if the distance function D and its conjugate D^* are sufficiently regular. For example, Lipschitz continuity of the functions $D^*(y,\cdot)$ (uniformly over $y \in Y$) and $\min_{\lambda\in\Delta_{n!}} D(A\lambda,\cdot)$ are sufficient for convergence of the two terms respectively. Also, note that in the static case when $p^t = p$, then the two error terms disappear.

Theorem 4.3 tells us that, for T sufficiently large, we are guaranteed to find an ϵ-approximate choice model λ in T iterations. The model λ will add weight $1/T$ on σ each time $\sigma = \sigma^t$ for $t \in [T]$. The number of iterations T depends on two factors: the convergence rate of online Mirror Descent, which requires $T \geq O(1/\epsilon^2)$; and the convergence rate of the two error terms, which depends on the Lipschitz constants as well as the rate that $p^t \to p$. Therefore, T upper-bounds the sparsity of the learned model λ as well. As a result, our analysis in essence exposes an explicit trade-off between the sparsity of the non-parametric choice model λ and its estimation accuracy ϵ. To the best of our knowledge, this explicit connection has not been characterized in the literature before.

Let us summarize the assumptions we have made on the general distance function D: to obtain a bounded domain $Y = \{y : \|y\| \leq R\}$, we need bounded gradients $\|\nabla_z D(z,p)\| \leq R$; to obtain the Mirror Descent bound, we need $\|a(\sigma^t) - \nabla_y D^*(y,p^t)\|_* \leq G$ for all $y \in Y$, and since $a(\sigma^t)$ are always bounded, we need a bound on $\|\nabla_y D^*(y,p^t)\|_*$; finally, to bound the error terms, we require Lipschitz continuity of $D^*(y,p)$ and $\min_{\lambda \in \Delta_{n!}} D(A\lambda, p)$ in the p variable.

Example 4.4. We examine when $D(A\lambda, p) = \|A\lambda - p\|$ is defined by a norm. Standard convex analysis results imply that $\|\nabla_y D(y,p)\|_* \leq 1$, hence we can define $Y = \{y : \|y\|_* \leq 1\}$ in terms of the dual norm. Also, $D^*(y,p) = \langle y, p\rangle$ when $\|y\|_* \leq 1$ and ∞ otherwise, thus $\nabla_y D^*(y,p^t) = p^t$ for $y \in Y$, and is bounded for any norm, since the data vectors p^t are bounded. Thus the regret bound from Lemma 4.2 can be applied, and $\sqrt{2\Omega G^2/T} \to 0$ as $T \to \infty$. Furthermore, $D^*(y,p^t) - D^*(y,p) = \langle y, p^t - y\rangle \leq \|p^t - p\|$ for any $y \in Y$, and from standard analysis results we know that $\min_{\lambda \in \Delta_{n!}} D(A\lambda, p^t) - \min_{\lambda \in \Delta_{n!}} D(A\lambda, p) \leq \|p^t - p\|$, thus the two error terms are bounded by $\frac{2}{T}\sum_{t=1}^{T} \|p^t - p\|$, which converges to 0 since $p^t \to p$. This shows that when D is defined any norm (in particular the ℓ_1-norm from Bertsimas and Mišic [5]), we can estimate a non-parametric choice model from a continuously updated sequence of data $p^t \to p$ efficiently. ∎

The main drawback of the Mirror Descent method is that each iteration t involves computing supergradients $\nabla_y f(y^t, p^T) = a(\sigma^t) - \nabla_y D^*(y^t, p^t)$, where $\sigma^t \in \arg\min_{\sigma' \in S_n} \langle a(\sigma'), y^t\rangle$. More precisely, in order to compute σ^t, we must solve the following subgradient generating subproblem at each iteration for the given vector y^t:

$$\min_{\sigma} \left\{ \sum_{j \in [m]} \sum_{i \in \mathcal{A}_j} y^t_{ij} a_{ij}(\sigma) : \ \sigma \in S_n \right\}. \tag{MD-SG(t)}$$

This problem has exactly the same form as the combinatorial subproblems identified in (RM-RC(β, ν)), (MLE-CG(S)) and (NM-CG(S)). In the next section, we will analyze this combinatorial subproblem more closely.

5 Combinatorial Subproblem

In this section, we take a closer look at the combinatorial subproblem encountered in (MLE-CG(S)), (NM-CG(S)) and (MD-SG(t)). Given a vector $y \in \mathbb{R}^N$, this problem takes the following general form

$$\max_{\sigma} \left\{ \langle y, a(\sigma)\rangle = \sum_{j \in [m]} \sum_{i \in \mathcal{A}_j} y_{ij} a_{ij}(\sigma) : \ \sigma \in S_n \right\}. \tag{14}$$

This problem has the following interpretation. Recall that $a(\sigma)$ is the column of the matrix A corresponding to the ranking σ. For each assortment $\mathcal{A}_j$, $j \in [m]$, $a(\sigma)$ essentially picks out the

item i in $\mathcal{A}_j$ that is ranked highest by σ. Therefore, when we choose a ranking σ, we go through each assortment $\mathcal{A}_j$ and pick out its highest ranked item $i_j(\sigma) = \arg\min_{i\in\mathcal{A}_j} \sigma(i)$. Then the objective value is $\langle y, a(\sigma)\rangle = \sum_{j\in[m]} y_{i_j(\sigma),j}$, i.e., we sum up the entries from the vector y corresponding to item $i_j(\sigma)$ in $\mathcal{A}_j$. The combinatorial subproblem (14) is then to choose the ranking σ that will give us the highest such sum.

5.1 NP-Hardness

Our problem (14) is closely related to a problem studied in van Ryzin and Vulcano [29]. They have shown in [29, Proposition 3] that the maximum weighted independent set problem, a strongly NP-hard problem, can be reduced to their problem. Essentially the same proof carries through for our problem, and we refer to [29, Proposition 3] for full details.

Theorem 5.1 ([29, Proposition 3])**.** *Every instance of a maximum weighted independent set problem on a graph $G = (V, E)$ with vertex weights w_i, $i \in V$, can be reduced to an instance of (14) in polynomial time. Thus, problem (14) is NP-hard.*

Problem (14) can also be seen as a generalization of the linear ordering problem, another known NP-hard problem, which involves choosing the best ordering of items $[n]$ to maximize *pairwise comparison scores* of the items $\{y_{ii'}\}_{i,i'\in[n]}$. In light of this, it is unlikely that there is an efficient algorithm to solve (14) exactly for arbitrary sets $\mathcal{A}_1, \ldots, \mathcal{A}_m$. Therefore, we instead look for useful special cases that can be solved efficiently.

Henceforth, we often talk about a column $a(\sigma)$ of a *fixed, arbitrary* ranking σ. Therefore, to simplify our exposition in the following sections, we drop the notation $a(\sigma)$ and $a_{ij}(\sigma)$ and simply write a and a_{ij} instead.

5.2 Integer Programming Formulations

We begin with some integer programming formulations of (14) that identify the feasible solutions via the set of possible incidence vectors $a(\sigma)$ for some ranking $\sigma \in S_n$.

A general formulation for (14) recently suggested by Bertsimas and Mišic [5] employs binary variables $z_{ii'}$ to encapsulate a ranking: specifically, $z_{ii'} = 1$ if and only if i is ranked before i'. For z to define a consistent ranking, we need to impose two extra types of constraints: $z_{ii'}+z_{i'i} = 1$, which ensures that either i is ranked below i', or vice versa, but not both or neither; and $z_{ii'}+z_{i'i''}-z_{ii''} \leq 1$, which ensures that if i is ranked before i', and i' is ranked before i'', then i must be ranked before i''. Then the incidence vector a is described via the variables $z_{ii'}$ as follows. Recall that we want $a_{ij} = 1$ if and only if i is the highest ranked item in $\mathcal{A}_j$. This is equivalent to $a_{ij} = 1$ if and only if $z_{ii'} = 1$ for all $i' \in \mathcal{A}_j \setminus \{i\}$. Therefore, we need the constraints $a_{ij} \leq z_{ii'}$ for all $i' \in \mathcal{A}_j$. Consequently, the general formulation of Bertsimas and Mišic [5] is based on the following set of constraints:

$$\begin{aligned}
\sum_{i\in\mathcal{A}_j} a_{ij} &= 1, && \forall j \in [m] \\
a_{ij} &\leq z_{ii'}, && \forall i, i' \in \mathcal{A}_j, \quad \forall j \in [m] \\
z_{ii'} + z_{i'i} &= 1, && \forall i, i' \in [n] \\
z_{ii'} + z_{i'i''} - z_{ii''} &\leq 1, && \forall i, i', i'' \in [n] \\
a_{ij}, z_{i'i''} &\in \{0,1\}, && \forall i, i', i'' \in [n],\ j \in [m].
\end{aligned} \tag{15}$$

The constraint set (15) can capture a wide variety of structural restrictions on the subsets $\mathcal{A}_j$ if needed. On the other hand, it is shown in [11, Page 1201] that the system defined by the constraints on z is not integral for $n \geq 6$, thus (15) is not integral either. Nevertheless, Méndez-Díaz et al. [21] recently carried out a polyhedral analysis of this problem and suggested a branch-and-cut algorithm with improved solution time over the naïve CPLEX solver.

We next present a different formulation for the incidence vectors a under a mild structural assumption on the subsets $\mathcal{A}_j$, $j \in [m]$. In other words, we impose an assumption on the hypergraph structure of the subsets. To this end, we define the intersection graph.

Definition 5.1 (Intersection graph)**.** Let $I_{\mathcal{A}} = (V_I, E_I)$ be the underlying (undirected) *intersection graph* of subsets $\mathcal{A}_1, \ldots, \mathcal{A}_m$, that is, $V_I = [m]$ and a pair $(j, j') \in E_I$ if and only if $\mathcal{A}_j \cap \mathcal{A}_{j'} \neq \emptyset$.

Since we want to model the case when the 'no-buy' option 1 is present in every assortment $\mathcal{A}_j$, we state the following condition which we will assume throughout.

Condition 5.1. There exists a set of common items $\mathcal{C} \subseteq \mathcal{A}_j$ for all $j \in [m]$.

The main structural condition that we want to examine is the following.

Condition 5.2. The intersection graph $I_{\mathcal{A}}$ of subsets $\mathcal{A}_1 \setminus \mathcal{C}, \ldots, \mathcal{A}_m \setminus \mathcal{C}$ is a tree.

Condition 5.2 implies that any item $i \notin \mathcal{C}$ belongs to at most two distinct subsets $\mathcal{A}_j \setminus \mathcal{C}$ and $\mathcal{A}_{j'} \setminus \mathcal{C}$. This allows us to dispense of the auxiliary variables z used in (15).

Theorem 5.2. *Under Conditions 5.1 and 5.2, the following system describes the set of feasible vectors $a(\sigma)$ in (14):*

$$\begin{aligned} \sum_{i \in \mathcal{A}_j} a_{ij} &= 1, \quad \forall\ j \in [m] \\ a_{ij} - a_{ij'} \leq \sum_{l \in \mathcal{A}_j \setminus \mathcal{A}_{j'}} a_{lj} + \sum_{l \in \mathcal{A}_{j'} \setminus \mathcal{A}_j} a_{lj'}, &\quad \forall\ i \in \mathcal{A}_j \cap \mathcal{A}_{j'}, \quad j, j' \in [m] \\ a_{ij} &\in \{0,1\}, \quad \forall\ i \in \mathcal{A}_j, \quad j \in [m]. \end{aligned} \tag{16}$$

The constraints in (16) have the following interpretation. Recall that $a_{ij} = 1$ if i is chosen from subset $\mathcal{A}_j$. Thus, the constraints $\sum_{i \in \mathcal{A}_j} a_{ij} = 1$ ensure that one item is always chosen from each subset $\mathcal{A}_j$. The constraints $a_{ij} - a_{ij'} \leq \sum_{l \in \mathcal{A}_j \setminus \mathcal{A}_{j'}} a_{lj} + \sum_{l \in \mathcal{A}_{j'} \setminus \mathcal{A}_j} a_{lj'}$ govern how items in the intersection $\mathcal{A}_j \cap \mathcal{A}_{j'}$ are chosen. More precisely, suppose $i \in \mathcal{A}_j \cap \mathcal{A}_{j'}$ is chosen from $\mathcal{A}_j$. The item chosen from $\mathcal{A}_{j'}$ must either be in $\mathcal{A}_{j'} \setminus \mathcal{A}_j$ or $\mathcal{A}_j \cap \mathcal{A}_{j'}$. If it is the former case, then the constraint has no effect. However, if it is the latter case, then the item from $\mathcal{A}_{j'}$ *must* be i in order to be consistent with a ranking.

In the absence of Condition 5.2, (16) may not be correct. This is true for the simplest possible example of a 3-cycle, even when $\mathcal{C} = \emptyset$.

Example 5.3. Let the set of items be $\{1, 2, 3\}$, with subsets $\mathcal{A}_1 = \{1, 2\}$, $\mathcal{A}_2 = \{2, 3\}$, $\mathcal{A}_3 = \{1, 3\}$. Then the corresponding intersection graph of Condition 5.2 is a 3-cycle. We set $a_{11} = a_{22} = a_{33} = 1$ and all the rest of variables a_{ij} to 0. This satisfies all of the constraints in the formulation. However, it will not be consistent with any ranking. Suppose for contradiction that it is consistent with ranking σ. Since $a_{11} = a_{22} = 1$, and $1 \in \mathcal{A}_1 \setminus \mathcal{A}_2$, $2 \in \mathcal{A}_2 \cap \mathcal{A}_1$, we require $\sigma(1) < \sigma(2)$. Since

$a_{22} = a_{33} = 1$, and also $2 \in \mathcal{A}_2 \setminus \mathcal{A}_3$ and $3 \in \mathcal{A}_3 \cap \mathcal{A}_2$, we must have $\sigma(2) < \sigma(3)$. But, then $a_{33} = a_{11} = 1$, along with $3 \in \mathcal{A}_3 \setminus \mathcal{A}_1$ and $1 \in \mathcal{A}_1 \cap \mathcal{A}_3$ implies $\sigma(3) < \sigma(1)$. Therefore, these three requirements on σ form a contradiction to σ being a ranking. ∎

Example 5.3 can be generalized to cycles of arbitrary length, which in fact shows that Condition 5.2 is necessary for (16). Even when $m = 2$, Condition 5.2 is insufficient to ensure the integrality of the LP relaxation of (16). This is demonstrated by the following example.

Example 5.4. Let the item set be $\{1, 2, 3, 4\}$. Consider the sets $\mathcal{A}_1 = \{1, 2, 3\}$ and $\mathcal{A}_2 = \{2, 3, 4\}$. Clearly, this set structure satisfies Condition 5.2 since there is just a single edge in the intersection graph $I_{\mathcal{A}}$. In this case, the solution $a_{21} = a_{22} = 1/3$, $a_{31} = a_{42} = 2/3$, $a_{11} = a_{32} = 0$. satisfies all of the constraints in (16) except the integrality restrictions. Moreover, this particular solution is a vertex of the LP relaxation of (16). ∎

5.3 Structural Assumptions Ensuring Efficient Solution Procedures

While Example 5.4 shows that Condition 5.2 does not in general admit integral vertices for the LP relaxation of (16), the following special case admits an efficient solution.

5.3.1 Disjoint Case

Our first condition examines a particular disjointness structure on assortments.

Condition 5.3. The sets $\mathcal{A}_1 \setminus \mathcal{C}, \ldots, \mathcal{A}_m \setminus \mathcal{C}$ are disjoint.

Let us label the items in $\mathcal{C}$ throughout this section as $\mathcal{C} = \{1, \ldots, k\} = [k]$.

Theorem 5.5. *Assume that Conditions 5.1 and 5.3 holds. Then formulation (16) has Chvatal rank at most k, and an exact LP formulation involving the rank k inequalities is given by*

$$\begin{aligned} \sum_{i=1}^{k} a_{ij} + \sum_{i=k+1}^{|\mathcal{A}_j|} a_{ij} &= 1, \quad \forall j \in [m] \\ a_{1j_1} + \ldots + a_{kj_k} &\leq 1, \quad \forall j_1, \ldots, j_k \in [m] \text{ (not necessarily distinct)} \\ a &\geq 0. \end{aligned} \tag{17}$$

Under Condition 5.3, an implication of Theorem 5.5 is that solving (14) is equivalent to solving a maximum weight independent set problem in a complete k-partite graph, and since complete k-partite graphs are perfect, we can do so by solving a LP with inequalities (17). However, because (17) involves m^k inequalities, this may be inefficient for large k values. Fortunately, due to the underlying graph structure, a maximum weight independent set problem in a complete k-partite graph can be solved much more efficiently. We describe the method in the context of our combinatorial subproblem (14) under Condition 5.3. Consider a set $\mathcal{A}_j$ for $j \in [m]$. The idea is to test out each $i \in \mathcal{C}$. For a fixed $i \in \mathcal{C}$, we set $a_{ij} = 1$ if and only if $y_{ij} > \max_{i' \in \mathcal{A}_j \setminus \mathcal{C}} y_{i'j}$, and otherwise set $a_{i'j} = 1$ where $i' = \arg\max_{i' \in \mathcal{A}_j \setminus \mathcal{C}} y_{i'j}$. Doing this for all $j \in [m]$ gives us a score for that particular $i \in \mathcal{C}$, and then we simply choose the highest score amongst the $i \in \mathcal{C}$. The running time of this method is $O(km)$, plus a one-off preprocessing to compute $i' = \arg\max_{i' \in \mathcal{A}_j \setminus \mathcal{C}} y_{i'j}$ for each $j \in [m]$, which takes $O(N)$ time in total.

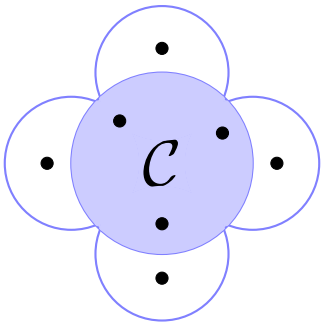

Figure 1: Illustration of Condition 5.3.

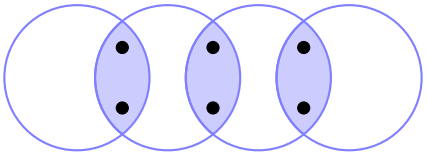

Figure 2: Illustration of Condition 5.4 ($\mathcal{C}$ not shown).

5.3.2 Path Case

A more reasonable condition on the intersection graph $I_{\mathcal{A}}$ than Condition 5.3 is that $I_{\mathcal{A}}$ is a simple path, i.e., a tree where each node has degree at most 2. This may arise, for instance, when a firm has an inherent order in its list of products, and clusters them into assortments according to this order.

Observation 5.6. Our first observation is that it is enough to consider the case when $|\mathcal{C}| = 1$. To see this, suppose we have $|\mathcal{C}| = k \geq 2$. Then for each item $i \in \mathcal{C}$, if i is chosen from $\mathcal{A}_j$ for some j, then for any other j', either i is chosen from $\mathcal{A}_{j'}$ or an item from $\mathcal{A}_{j'} \setminus \mathcal{C}$ is chosen. Choosing a different item from $\mathcal{C}$ in some set j' will contradict the ranking structure. Thus, we can just solve the case when the common set is simply $\{i\}$, record the score, then repeat this for all $i \in \mathcal{C}$ to get the best score. ∎

Henceforth, we will work with the following condition.

Condition 5.4. The intersection graph $I_{\mathcal{A}}$ is a path, and $\mathcal{C} = \{1\}$ is a singleton.

Since indexing does not matter, we will assume without loss of generality that the path is from 1 to m, that is, $\mathcal{A}_j \cap \mathcal{A}_{j'} \neq \emptyset$ if and only if $|j - j'| \leq 1$.

Under Condition 5.4, we present an efficient algorithm for (14) via recursion. Let us first define some notation. Given a fixed y vector, we define:

- $\mathcal{V}(\mathcal{A}_j^j) := \max_{i \in \mathcal{A}_j} y_{ij}$. In other words, $\mathcal{V}(\mathcal{A}_j^j)$ is the optimal solution of (14) on a single subset $\mathcal{A}_j$, using scores from the segment of y corresponding to $\mathcal{A}_j$ (i.e., $\{y_{ij}\}_{i \in \mathcal{A}_j}$).
- $\mathcal{V}(\mathcal{A}_{j \setminus j'}^j) := \max_{i \in \mathcal{A}_j \setminus \mathcal{A}_{j'}} y_{ij}$. In other words, $\mathcal{V}(\mathcal{A}_{j \setminus j'}^j)$ is the optimal solution of (14) on $\mathcal{A}_j \setminus \mathcal{A}_{j'}$, using scores from the $\mathcal{A}_j$ segment of y.
- $\mathcal{V}(\mathcal{A}_{j \cap j'}^j) := \max_{i \in \mathcal{A}_j \cap \mathcal{A}_{j'}} y_{ij}$.
- $\mathcal{V}(\mathcal{A}_{(j \cap j') \setminus \mathcal{C}}^+) := \max_{i \in (\mathcal{A}_j \cap \mathcal{A}_{j'}) \setminus \mathcal{C}} \{y_{ij} + y_{ij'}\}$. In other words, $\mathcal{V}(\mathcal{A}_{j \cap j'}^+)$ is the optimal of solution of (14) when we have two *identical subsets*, both $(\mathcal{A}_j \cap \mathcal{A}_{j'}) \setminus \mathcal{C}$, but with different scores for each, $\{y_{ij}\}_{i \in (\mathcal{A}_j \cap \mathcal{A}_{j'}) \setminus \mathcal{C}}$ and $\{y_{ij'}\}_{i \in (\mathcal{A}_j \cap \mathcal{A}_{j'}) \setminus \mathcal{C}}$. Since the items chosen from each subset must be identical, the optimal solution is the item with the highest sum.

- $\mathcal{V}(\mathcal{A}_{1:m})$ corresponds to the optimal solution of (14) on subsets $\mathcal{A}_1, \ldots, \mathcal{A}_m$.
- $\mathcal{V}(\mathcal{A}_{(1:m-1\setminus m)\cup\mathcal{C}})$ corresponds to the optimal solution of (14) on subsets $(\mathcal{A}_1\setminus\mathcal{A}_m)\cup\mathcal{C}, \ldots, (\mathcal{A}_{m-1}\setminus\mathcal{A}_m)\cup\mathcal{C}$. Note that given the path structure, the sets $(\mathcal{A}_1 \setminus \mathcal{A}_m)\cup\mathcal{C}, \ldots, (\mathcal{A}_{m-1} \setminus \mathcal{A}_m)\cup\mathcal{C}$ are respectively the same as the sets $\mathcal{A}_1, \ldots, \mathcal{A}_{m-2}, (\mathcal{A}_{m-1} \setminus \mathcal{A}_m)\cup\mathcal{C}$, so only $\mathcal{A}_{m-1}$ is affected.

Our strategy is to show that a dynamic programming formulation to compute the optimal value $\mathcal{V}(\mathcal{A}_{1:m})$ is efficient under Condition 5.4. The first step is to give a recursive formulation of $\mathcal{V}(\mathcal{A}_{1:m})$, which we present as a lemma.

Lemma 5.7. *Under Condition 5.4, the following recursive formula holds:*

$$\mathcal{V}(\mathcal{A}_{1:m}) = \max \begin{cases} \mathcal{V}(\mathcal{A}_{m\setminus m-1}) + \mathcal{V}(\mathcal{A}_{1:m-1}) \\ \mathcal{V}(\mathcal{A}^m_{m\cap m-1}) + \mathcal{V}(\mathcal{A}_{(1:m-1\setminus m)\cup\mathcal{C}}) \\ \mathcal{V}(\mathcal{A}^+_{m\cap m-1}) + \mathcal{V}(\mathcal{A}_{1:m-2}) \end{cases} . \tag{18}$$

Finally, due to some efficient simplifications, the recursive formula from Lemma 5.7 can be used to compute $\mathcal{V}(\mathcal{A}_{1:m})$ efficiently.

Theorem 5.8. *Under Condition 5.4, the running time of the recursive method is $O(N)$. In the case when Condition 5.4 holds except for $|\mathcal{C}| = k > 1$, then the running time is $O(kN)$.*

6 Conclusion and Future Directions

In this paper, we presented a general framework to dynamically estimate non-parametric choice models from online data using tools from convex conjugacy, online convex optimization, and joint estimation-optimization framework. Our framework unifies a number of previous models used in the non-parametric choice estimation literature. Moreover, the solution approaches presented in the prior literature to estimate choice models from the same type of data did not come with convergence rate analysis or allow for dynamic and efficient updates when data is continuously updated. As opposed to the prior literature, our approach both enjoys provably efficient iteration guarantees and also converges even when the data is dynamically updated each iteration. In addition, we studied a common NP-hard combinatorial problem which arises in our framework as well as all other non-parametric estimation methods. We identified a number of structural assumptions on the assortments $\mathcal{A}_j$ where this problem can be solved in polynomial-time. Together with our efficiency guarantees for our general framework, this establishes a completely efficient framework for estimating choice models under certain structural assumptions.

A number of research avenues are worthy of further investigation. In Farias et al. [10] and van Ryzin and Vulcano [29], guarantees for the recovery of the correct choice model were proved for the revenue maximization and MLE approaches respectively. The question of whether similar correct recovery guarantees exist our more general framework is appealing. In particular, [29, Corollary 1] states that as we get more accurate data $p^t \to p$, the MLE $\hat{\lambda}^t$, i.e., the solution to (6) with the data p^t, converges to the true non-parametric choice model λ with probability 1. However, $\hat{\lambda}^t$ is the full solution to (6). It would be nice to extend this to our more general approach, that is, obtain the same recovery guarantee $\hat{\lambda}^t \to \lambda$ with high probability when $p^t \to p$, with the crucial difference being that the $\hat{\lambda}^t$ are cheaply updated from previous time steps according to our simple online updates rather than being full solutions of (9) for each data point p^t.

For the combinatorial problem (14), we conjecture that it is possible to obtain a polynomial-time algorithm for the more general Condition 5.2 without imposing Condition 5.4. This could be done via an efficient recursive formulation as in Section 5.3. Identification of other or more general structural assumptions than Condition 5.2 that ensure polynomial-time algorithms is of interest.

Finally, Honhon et al. [13] exhibit efficient algorithms for various structural assumptions on customer types. In a similar vein, it is compelling to investigate whether our structural assumptions on the subsets allow us to efficiently solve the associated assortment optimization problem as well.

Acknowledgments

This research is supported in part by NSF grant CMMI 1454548.

References

[1] S. Agrawal and N.R. Devanur. Fast algorithms for online stochastic convex programming. In *Proceedings of the Twenty-sixth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1405–1424, Philadelphia, PA, USA, 2015. SIAM.

[2] H. Ahmadi and U. V. Shanbhag. Data-driven first-order methods for misspecified convex optimization problems: Global convergence and rate estimates. In *53rd IEEE Conference on Decision and Control*, pages 4228–4233, Dec 2014.

[3] A. Aouad, V.F. Farias, R. Levi, and D. Segev. The approximability of assortment optimization under ranking preferences. Technical report, June 2015. URL https://ssrn.com/abstract=2612947.

[4] M.E. Ben-Akiva and S.R. Lerman. *Discrete Choice Analysis: Theory and Application to Travel Demand*, volume 9. MIT Press, 1985.

[5] D. Bertsimas and V.V. Mišic. Data-driven assortment optimization. 2015.

[6] H.D. Block and J. Marschak. Random orderings and stochastic theories of responses. *Contributions to Probability and Statistics*, 2:97–132, 1960.

[7] G. Calafiore and M.C. Campi. Uncertain convex programs: randomized solutions and confidence levels. *Mathematical Programming*, 102(1):25–46, 2005.

[8] N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, New York, NY, USA, 2006. ISBN 0521841089.

[9] V.F. Farias, S. Jagabathula, and D. Shah. Sparse choice models. In *Information Sciences and Systems (CISS), 2012 46th Annual Conference on*, pages 1–28. IEEE, 2012.

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

[11] M. Grötschel, M. Jünger, and G. Reinelt. A cutting plane algorithm for the linear ordering problem. *Operations Research*, 32(6):1195–1220, 1984.

[12] N. Ho-Nguyen and F. Kılınç-Karzan. Accelerating optimization under uncertainty via online convex optimization. Technical report, August 2016. http://www.optimization-online.org/DB_HTML/2016/08/5571.html.

[13] D. Honhon, S. Jonnalagedda, and X.A. Pan. Optimal algorithms for assortment selection under ranking-based consumer choice models. *Manufacturing & Service Operations Management*, 14 (2):279–289, 2012.

[14] A. Juditsky and A. Nemirovski. First-order methods for nonsmooth convex large-scale optimization, I: General purpose methods. In S. Sra, S. Nowozin, and S.J. Wright, editors, *Optimization for Machine Learning*, Neural information processing series. MIT Press, 2012.

[15] A. Juditsky and A. Nemirovski. First-order methods for nonsmooth convex large-scale optimization, ii: Utilizing problem's structure. In S. Sra, S. Nowozin, and S.J. Wright, editors, *Optimization for Machine Learning*, Neural information processing series. MIT Press, 2012.

[16] A. Juditsky, F. Kılınç-Karzan, and A. Nemirovski. Randomized first order algorithms with applications to ℓ_1 minimization. *Mathematical Programming*, 142(1-2):269–310, 2013.

[17] A.G. Kök, M.L. Fisher, and R. Vaidyanathan. *Assortment Planning: Review of Literature and Industry Practice*, pages 175–236. Springer US, Boston, MA, 2015.

[18] S. Mahajan and G. van Ryzin. Stocking retail assortments under dynamic consumer substitution. *Operations Research*, 49(3):334–351, 2001.

[19] D. McFadden. Modeling the choice of residential location. *Transportation Research Record*, (673), 1978.

[20] D. McFadden and K. Train. Mixed MNL models for discrete response. *Journal of Applied Econometrics*, 15(5):447–470, 2000.

[21] I. Méndez-Díaz, G. Vulcano, and P. Zabala. Analysis of a generalized linear ordering problem via integer programming. Technical report, July 2016. http://pages.stern.nyu.edu/~gvulcano/GLOP.pdf.

[22] V. Mirrokni, R.P. Leme, A. Vladu, and S.C.-W. Wong. Tight bounds for approximate Carathéodory and beyond. *arXiv Preprint 1512.08602*, 2015.

[23] A. Nemirovski. Prox-method with rate of convergence $O(1/t)$ for variational inequalities with Lipschitz continuous monotone operators and smooth convex-concave saddle point problems. *SIAM Journal on Optimization*, 15:229–251, 2004.

[24] M.R. Ratliff, B. V. Rao, P.C. Narayan, and K. Yellepeddi. A multi-flight recapture heuristic for estimating unconstrained demand from airline bookings. *Journal of Revenue and Pricing Management*, 7(2):153–171, 2008.

[25] P. Rusmevichientong, B. Van Roy, and P.W. Glynn. A nonparametric approach to multiproduct pricing. *Operations Research*, 54(1):82–98, 2006.

[26] M. Sion. On general minimax theorems. *Pacific J. Math.*, 8(1):171–176, 1958. URL http://projecteuclid.org/euclid.pjm/1103040253.

[27] K. Talluri and G. van Ryzin. *The theory and practice of revenue management*, volume 68 of *International Series in Operations Research & Management Science*. Springer US, 2004.

[28] G. van Ryzin and S. Mahajan. On the relationship between inventory costs and variety benefits in retail assortments. *Management Science*, 45(11):1496–1509, 1999.

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

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

[31] H.C.W.L. Williams. On the formation of travel demand models and economic evaluation measures of user benefit. *Environment and Planning A*, 9(3):285–344, 1977.

A Proofs

Proof of Lemma 4.2. This follows directly from Theorem 4.1. □

Proof of Theorem 4.3. Note that we have

$$
\begin{aligned}
&D\left(\frac{1}{T}\sum_{t=1}^{T} a(\sigma^t),\, p\right) \\
&= \max_{y\in Y}\left\{\left\langle \frac{1}{T}\sum_{t=1}^{T} a(\sigma^t), y\right\rangle - D^*(y,p)\right\} \\
&= \max_{y\in Y}\frac{1}{T}\sum_{t=1}^{T}\left[\langle a(\sigma^t), y\rangle - D^*(y,p^t) + D^*(y,p^t) - D^*(y,p)\right] \\
&= \max_{y\in Y}\frac{1}{T}\sum_{t=1}^{T}\left[f^t(y) + D^*(y,p^t) - D^*(y,p)\right] \\
&\le \max_{y\in Y}\frac{1}{T}\sum_{t=1}^{T} f^t(y^t) - \frac{1}{T}\sum_{t=1}^{T} f^t(y^t) + \frac{1}{T}\sum_{t=1}^{T} f^t(y^t) + \max_{y\in Y}\frac{1}{T}\sum_{t=1}^{T}\left[D^*(y,p^t) - D^*(y,p)\right]. \\
&\le \sqrt{\frac{2\Omega L^2}{T}} + \frac{1}{T}\sum_{t=1}^{T} f^t(y^t) + \max_{y\in Y}\frac{1}{T}\sum_{t=1}^{T}\left[D^*(y,p^t) - D^*(y,p)\right],
\end{aligned}
$$

where the first inequality follows from decomposing the terms in the maximum, and the last inequality follows from Lemma 4.2. Furthermore, we have $f^t(y^t) = \langle y^t, a(\sigma^t)\rangle - D^*(y^t;p^t) = f(y^t;p^t)$, hence

$$
f^t(y^t) = f(y^t;p^t) \le \max_{y\in Y} f(y,p^t) = \max_{y\in Y}\min_{\lambda\in\Delta_{n!}}\left\{\langle A\lambda, y\rangle - D^*(y,p^t)\right\} = \min_{\lambda\in\Delta_{n!}} D(A\lambda,p^t).
$$

Therefore,

$$
\begin{aligned}
&D\left(\frac{1}{T}\sum_{t=1}^{T} a(\sigma^t), p\right) - \min_{\lambda\in\Delta_{n!}} D(A\lambda,p) \\
&\le \sqrt{\frac{2\Omega L^2}{T}} + \max_{y\in Y}\frac{1}{T}\sum_{t=1}^{T}\left[D^*(y,p^t) - D^*(y,p)\right] + \frac{1}{T}\sum_{t=1}^{T}\left[\min_{\lambda\in\Delta_{n!}} D(A\lambda,p^t) - \min_{\lambda\in\Delta_{n!}} D(A\lambda,p)\right].
\end{aligned}
$$

□

The following lemma is key to proving Theorem 5.2.

Lemma A.1. *Let the vector a correspond to a column in A associated with some ranking σ. Suppose there exists distinct $j, j' \in [m]$ and $i_j \in \mathcal{A}_j$, $i_{j'} \in \mathcal{A}_{j'}$ such that $a_{i_j j} = a_{i_{j'} j'} = 1$ and $i_{j'} \in \mathcal{A}_j \cap \mathcal{A}_{j'}$. If $i_j \in \mathcal{A}_j \setminus \mathcal{A}_{j'}$, then we must have $\sigma(i_j) < \sigma(i_{j'})$. If $i_j \in \mathcal{A}_j \cap \mathcal{A}_{j'}$ also, then $i_j = i_{j'}$.*

Proof. Suppose that $i_j \in \mathcal{A}_j \setminus \mathcal{A}_{j'}$, in particular this means $i_j \neq i_{j'}$, and for contradiction suppose that $\sigma(i_j) \ge \sigma(i_{j'})$. Since $i_j \neq i_{j'}$, $\sigma(i_j) > \sigma(i_{j'})$. Since $a_{i_j j} = 1$, $i_j = \arg\min_{i\in\mathcal{A}_j}\sigma(i)$. But $i_{j'} \in \mathcal{A}_j$ also, and $\sigma(i_{j'}) < \sigma(i_j) = \arg\min_{i\in\mathcal{A}_j}\sigma(i)$, a contradiction.

Now suppose $i_j, i_{j'} \in \mathcal{A}_j \cap \mathcal{A}_{j'}$. Suppose for contradiction that $i_j \neq i_{j'}$ and (without loss of generality) $\sigma(i_j) > \sigma(i_{j'})$. Since $a_{i_j j} = 1$, we have $\sigma(i_j) = \arg\min_{i \in \mathcal{A}_j} \sigma(i) < \sigma(i_{j'})$ because $i_{j'} \in \mathcal{A}_j$, and hence this is a contradiction. Thus, $i_j = i_{j'}$. □

Proof of Theorem 5.2. Let the vector a correspond to a column in A associated with some ranking σ. Clearly the constraints $\sum_{i \in \mathcal{A}_j} a_{ij} = 1$ and $a_{ij} \in \{0, 1\}$ are satisfied. Consider now two subsets $\mathcal{A}_j, \mathcal{A}_{j'}$ with non-empty intersection. Suppose that $a_{ij} - a_{ij'} \leq \sum_{l \in \mathcal{A}_j \setminus \mathcal{A}_{j'}} a_{lj} + \sum_{l \in \mathcal{A}_{j'} \setminus \mathcal{A}_j} a_{lj'}$ is not satisfied for some $i \in \mathcal{A}_j \cap \mathcal{A}_{j'}$. This means that $a_{ij} = 1$, $a_{ij'} = 0$, $a_{lj} = 0$ for $l \in \mathcal{A}_j \setminus \mathcal{A}_{j'}$ and $a_{lj'} = 0$ for $l \in \mathcal{A}_{j'} \setminus \mathcal{A}_j$. Since $\sum_{i \in \mathcal{A}_{j'}} a_{ij'} = 1$, there must exist some $i' \in \mathcal{A}_j \cap \mathcal{A}_{j'}$, $i' \neq i$ such that $a_{i'j'} = 1$. This is contrary to Lemma A.1, which states that $i, i' \in \mathcal{A}_j \cap \mathcal{A}_{j'}$, $a_{ij} = a_{i'j'} = 1$ implies $i = i'$.

Conversely, given some vector a feasible with respect to (16), we want to find a ranking σ consistent with a. Let $i_j \in \mathcal{A}_j$ be the index for which $a_{i_j j} = 1$, for $j \in [m]$. We propose the ranking σ of $i_1, \ldots, i_m$ at the top in some order to be determined (removing repeats if necessary), and ordering the other items $[n] \setminus \{i_1, \ldots, i_m\}$ after in any order. For each $j \in [m]$, the order we determine needs to satisfy $i_j = \arg\min_{i \in \mathcal{A}_j} \sigma(i)$.

To see how to do this, fix some j. If there does not exist $j' \neq j$ such that $i_{j'} \in \mathcal{A}_j$, then the order of i_j relative to the other $i_{j'}$ does not matter. Suppose now that there exists some $j' \neq j$ such that $i_{j'} \in \mathcal{A}_j$. If $i_j \in \mathcal{A}_{j'}$ also, then we must have $i_j = i_{j'}$ from Lemma A.1. If $i_j \in \mathcal{A}_j \setminus \mathcal{A}_{j'}$, then from Lemma A.1 we must have $\sigma(i_j) < \sigma(i_{j'})$. Thus, the problem is the following: we want to order $i_1, \ldots, i_m$ in some fashion so that $\sigma(i_j) < \sigma(i_{j'})$ whenever $i_j \in \mathcal{A}_j \setminus \mathcal{A}_{j'}$ and $i_{j'} \in \mathcal{A}_j \cap \mathcal{A}_{j'}$.

To do this, we consider the directed graph $D = (V, E)$ where vertices are $V = \{i_j : j \in [m]\}$, and *directed* edge $(i_j, i_{j'}) \in E$ whenever $i_j \in \mathcal{A}_j \setminus \mathcal{A}_{j'}$ and $i_{j'} \in \mathcal{A}_j \cap \mathcal{A}_{j'}$. Note that there are no anti-parallel edges in D. The idea is to find some ordering of V so that $i_j < i_{j'}$ whenever $(i_j, i_{j'}) \in E$. This is exactly the condition needed for σ. We can employ a *topological sorting* algorithm on D to find our desired order, and thus completing σ. Before continuing, we argue that given such an ordering, our constructed σ indeed gives us the right incidence vector. Fix a $j \in [m]$. For $i \in \mathcal{A}_j \setminus V$, we clearly have $\sigma(i_j) < \sigma(i)$. For $i_{j'} \in V$ with $i_{j'} \in \mathcal{A}_j$, the first case is $i_j \in \mathcal{A}_{j'}$, implying that $i_j = i_{j'}$ by the constraints on a, and hence $\sigma(i_j) = \sigma(i_{j'})$. The second case is $i_j \notin \mathcal{A}_{j'}$, which means that $\sigma(i_j) < \sigma(i_{j'})$ by definition of our constructed ordering. Thus, $i_j = \arg\min_{i \in \mathcal{A}_j} \sigma(i)$. This proves that σ indeed gives the correct incidence vector.

The last thing we need to address is whether or not we may find a topological ordering on the directed graph D. This will be possible if the graph D is acyclic. To see this, note that an edge $(i_j, i_{j'})$ is present in D only if $\mathcal{A}_j \cap \mathcal{A}_{j'} \neq \emptyset$. Furthermore, if $i_j \in \mathcal{C}$ for some j, then it cannot have any outgoing edges since $\mathcal{A}_j \setminus \mathcal{A}_{j'}$ never contains $\mathcal{C}$, as $\mathcal{C} \subseteq \mathcal{A}_{j'}$. By Condition 5.2, D cannot contain a directed cycle. To see why this is, suppose for contradiction that it does. By Condition 5.2, a cycle cannot be formed without items from $\mathcal{C}$. However, a cycle containing an item from $\mathcal{C}$ is a contradiction since it cannot have outgoing edges. Hence, we can find a topological ordering on D. □

Proof of Theorem 5.5. Our first observation is that we can without loss of generality assume that $\mathcal{A}_j \setminus \mathcal{C}$ contains exactly one element, which we will denote as j_0, with corresponding score $y_{j_0 j}$. To see this, first suppose that $|\mathcal{A}_j \setminus \mathcal{C}| \geq 2$ for some j. If the item chosen from subset $\mathcal{A}_j$ is from $\mathcal{A}_j \setminus \mathcal{C}$, then the best item will always be the one with highest score y_{ij} for $i \in \mathcal{A}_j \setminus \mathcal{C}$. Call this item j_0. By Condition 5.3, the other subsets do not prevent us from choosing j_0. Therefore, we can throw away

all items in $\mathcal{A}_j \setminus \mathcal{C}$ except j_0. Now suppose that $\mathcal{A}_j \setminus \mathcal{C} = \emptyset$. Then we can construct an artificial item j_0 and assign it score $y_{j_0 j} < y_{ij}$ for all $i \in \mathcal{C}$. This ensures that any optimal solution never chooses j_0.

Since we are assuming that $\mathcal{A}_j \setminus \mathcal{C} = \{j_0\}$, the first constraint in (16) implies that $a_{j_0 j} = 1 - \sum_{i \in \mathcal{C}} a_{ij}$. Then using this relation, we may remove the variables $a_{j_0 j}$, and obtain the following simplified formulation with only items from the common set $\mathcal{C}$:

$$2a_{ij} + \sum_{i' \in \mathcal{C}, i' \neq i} (a_{ij} + a_{ij'}) \leq 2, \quad \forall\ i \in \mathcal{C},\ j, j' \in [m] \tag{19}$$

$$a_{ij} \in \{0, 1\}. \quad \forall\ i \in \mathcal{C},\ j \in [m].$$

We first claim that the rank 1 inequalities are of the form $a_{ij_i} + a_{i'j_{i'}} \leq 1$ for $i \neq i'$. By symmetry, it is enough to show this for $i = 1$, $i' = 2$. Summing up the following two inequalities from (19):

$$2a_{1j_1} + \sum_{i \neq 1} (a_{ij_1} + a_{ij_2}) \leq 2$$

$$2a_{2j_2} + \sum_{i \neq 2} (a_{ij_1} + a_{ij_2}) \leq 2$$

leads to

$$3a_{1j_1} + 3a_{2j_2} + a_{2j_1} + a_{1j_2} + 2 \sum_{i \neq 1,2} (a_{ij_1} + a_{ij_2}) \leq 4.$$

Dividing both sides by 3 and performing the Chvatal-Gomory rounding procedure gives $a_{1j_1} + a_{2j_2} \leq 1$.

The three-term inequalities of the form $a_{1j_1} + a_{2j_2} + a_{3j_3} \leq 1$ can be obtained by performing the rounding procedure on the inequalities $a_{1j_1} + a_{2j_2} \leq 1$, $a_{1j_1} + a_{3j_3} \leq 1$, and $a_{2j_2} + a_{3j_3} \leq 1$. The right hand side of the sum will be 3, whereas each coefficient will be 2, and applying the rounding procedure gives us the desired inequality. These three-term inequalities are of rank 2.

In general, for $2 < k' \leq k$, inequalities of the form $a_{1j_1} + \ldots + a_{k'j_{k'}} \leq 1$ can be obtained through the $(k'-1)$-term inequalities. Specifically, add all $(k'-1)$-term inequalities from items $1, \ldots, k'$. There are $\binom{k'}{k'-1} = k'$ of these, so the right hand side will be k'. Furthermore, each term a_{ij_i} will appear in exactly $k'-1$ inequalities, so the coefficients will be $k'-1$. Dividing by $k'-1$, we can round the right hand side to $\lfloor \frac{k'}{k'-1} \rfloor = 1$. These inequalities are of rank $k'-1$.

The above argument implies that the inequalities (17) are of rank at most $k-1$. In fact, it is easy to see that the set of inequalities (17) includes those from (19), so henceforth in the proof we focus just on (17). Our aim now is to prove that (17), together with non-negativity constraints, (17) defines an integral polytope.

To see this, consider the intersection graph G defined by the columns of the coefficient matrix of (17). The vertices of G are simply the variables a_{ij}, $i \in [k]$, $j \in [m]$. There is an edge between a pair of vertices $a_{ij}, a_{i'j'}$ if and only if $i \neq i'$. However, notice now that G is a complete k-partite graph, with the partitions of vertices defined by the items: $P_i = \{a_{ij} : j \in [m]\}$. But this means that the coefficient matrix of (17) is simply the clique-node matrix of a complete k-partite graph. It is known that complete k-partite graphs are perfect, and thus the polytope defined by its clique matrix with right hand side vector of ones is integral. $\square$

Proof of Lemma 5.7. The idea of the recursion is to analyze which item from $\mathcal{A}_m$ will be chosen. Let this item be i_m, and similarly let the item from $\mathcal{A}_{m-1}$ be denoted by i_{m-1}. We divide our analysis into several cases.

Suppose first that $i_m \in \mathcal{A}_m \setminus \mathcal{A}_{m-1}$. Then the value is $\mathcal{V}(\mathcal{A}^m_{m\setminus m-1}) + \mathcal{V}(\mathcal{A}_{1:m-1})$. This is one of the terms in (18) and does not need further analysis.

Let us not consider the case where $i_m \in \mathcal{A}_m \cap \mathcal{A}_{m-1}$. There are several subcases to consider here.

- If $i_m \in (\mathcal{A}_m \cap \mathcal{A}_{m-1}) \setminus \mathcal{C}$, and $i_{m-1} \in (\mathcal{A}_m \cap \mathcal{A}_{m-1}) \setminus \mathcal{C}$ also, then we must have $i_m = i_{m-1}$, and the optimal value is $\mathcal{V}(\mathcal{A}^+_{(m\cap m-1)\setminus\mathcal{C}}) + \mathcal{V}(\mathcal{A}_{1:m-2})$. This is one of the terms in (18) and does not need further analysis.
- If $i_m \in (\mathcal{A}_m \cap \mathcal{A}_{m-1}) \setminus \mathcal{C}$, and $i_{m-1} \notin (\mathcal{A}_m \cap \mathcal{A}_{m-1}) \setminus \mathcal{C}$, then i_{m-1} can be anywhere in $(\mathcal{A}_{m-1} \setminus \mathcal{A}_m) \cup \mathcal{C}$, thus the optimal value is $\mathcal{V}(\mathcal{A}^m_{(m\cap m-1)\setminus\mathcal{C}}) + \mathcal{V}(\mathcal{A}_{(1:m-1\setminus m)\cup\mathcal{C}})$.
- If $i_m \in \mathcal{C}$, and $i_{m-1} \notin \mathcal{C}$, then i_{m-1} can be anywhere from $\mathcal{A}_{m-1} \setminus \mathcal{A}_m$, thus the optimal value is $\mathcal{V}(\mathcal{A}^m_{m\cap\mathcal{C}}) + \mathcal{V}(\mathcal{A}_{1:m-2}, \mathcal{A}_{m-1} \setminus \mathcal{A}_m)$. Note that there is no restriction on choosing items for $\mathcal{A}_{1:m-2}$ despite $i_m \in \mathcal{C}$ (which is in every set), because $\mathcal{A}_m$ and $\mathcal{A}_{1:m-2}$ have no other common items, and $\mathcal{C} = \{1\}$ is a singleton. Note that this reasoning would not work if $\mathcal{C}$ were not a singleton.
- If $i_m \in \mathcal{C}$ and $i_{m-1} \in \mathcal{C}$, then the item from $\mathcal{A}_{m-2}$ cannot be from $(\mathcal{A}_{m-2} \cap \mathcal{A}_{m-1}) \setminus \mathcal{C}$, hence the optimal value is $\mathcal{V}(\mathcal{A}^+_{m\cap m-1\cap\mathcal{C}}) + \mathcal{V}(\mathcal{A}_{(1:m-2\setminus m-1)\cup\mathcal{C}})$. Again, there is no restriction on choosing items for $\mathcal{A}_{1:m-3}$ for the same reasons as above. The only restriction is on the item chosen from $\mathcal{A}_{m-2}$, which is reflected in the term $\mathcal{V}(\mathcal{A}_{(1:m-2\setminus m-1)\cup\mathcal{C}})$.

With these observations, we have the equation

$$\mathcal{V}(\mathcal{A}_{1:m}) = \max \begin{cases} \mathcal{V}(\mathcal{A}^m_{m\setminus m-1}) + \mathcal{V}(\mathcal{A}_{1:m-1}) \\ \mathcal{V}(\mathcal{A}^+_{(m\cap m-1)\setminus\mathcal{C}}) + \mathcal{V}(\mathcal{A}_{1:m-2}) \\ \quad \max \begin{cases} \mathcal{V}(\mathcal{A}^m_{(m\cap m-1)\setminus\mathcal{C}}) + \mathcal{V}(\mathcal{A}_{(1:m-1\setminus m)\cup\mathcal{C}}) \\ \mathcal{V}(\mathcal{A}^m_{m\cap\mathcal{C}}) + \mathcal{V}(\mathcal{A}_{1:m-2}, \mathcal{A}_{m-1} \setminus \mathcal{A}_m) \\ \mathcal{V}(\mathcal{A}^+_{m\cap m-1\cap\mathcal{C}}) + \mathcal{V}(\mathcal{A}_{(1:m-2\setminus m-1)\cup\mathcal{C}}). \end{cases} \end{cases}$$

It remains to show that

$$\mathcal{V}(\mathcal{A}^m_{m\cap m-1}) + \mathcal{V}(\mathcal{A}_{(1:m-1\setminus m)\cup\mathcal{C}}) = \max \begin{cases} \mathcal{V}(\mathcal{A}^m_{(m\cap m-1)\setminus\mathcal{C}}) + \mathcal{V}(\mathcal{A}_{(1:m-1\setminus m)\cup\mathcal{C}}) \\ \mathcal{V}(\mathcal{A}^m_{m\cap\mathcal{C}}) + \mathcal{V}(\mathcal{A}_{1:m-2}, \mathcal{A}_{m-1} \setminus \mathcal{A}_m) \\ \mathcal{V}(\mathcal{A}^+_{m\cap m-1\cap\mathcal{C}}) + \mathcal{V}(\mathcal{A}_{(1:m-2\setminus m-1)\cup\mathcal{C}}). \end{cases}$$

We observe that

$$\mathcal{V}(\mathcal{A}^m_{m\cap m-1}) = \max \begin{cases} \mathcal{V}(\mathcal{A}^m_{(m\cap m-1)\setminus\mathcal{C}}) \\ \mathcal{V}(\mathcal{A}^m_{m\cap\mathcal{C}}), \end{cases}$$

thus

$$\mathcal{V}(\mathcal{A}^m_{m\cap m-1}) + \mathcal{V}(\mathcal{A}_{(1:m-1\setminus m)\cup\mathcal{C}}) = \max \begin{cases} \mathcal{V}(\mathcal{A}^m_{(m\cap m-1)\setminus\mathcal{C}}) + \mathcal{V}(\mathcal{A}_{(1:m-1\setminus m)\cup\mathcal{C}}) \\ \mathcal{V}(\mathcal{A}^m_{m\cap\mathcal{C}}) + \mathcal{V}(\mathcal{A}_{(1:m-1\setminus m)\cup\mathcal{C}}). \end{cases}$$

Now observe that by analyzing the cases $i_{m-1} \in \mathcal{C}$ and $i_{m-1} \notin \mathcal{C}$, we get

$$\mathcal{V}(\mathcal{A}_{(1:m-1\setminus m)\cup\mathcal{C}}) = \max\begin{cases}\mathcal{V}(\mathcal{A}^{m-1}_{m-1\cap\mathcal{C}}) + \mathcal{V}(\mathcal{A}_{(1:m-2\setminus m-1)\cup\mathcal{C}})\\ \mathcal{V}(\mathcal{A}_{1:m-3}, \mathcal{A}_{m-2}\setminus\mathcal{A}_{m-1}).\end{cases}$$

But now this implies that

$$\begin{aligned}\mathcal{V}(\mathcal{A}^m_{m\cap m-1}) + \mathcal{V}(\mathcal{A}_{(1:m-1\setminus m)\cup\mathcal{C}}) &= \max\begin{cases}\mathcal{V}(\mathcal{A}^m_{(m\cap m-1)\setminus\mathcal{C}}) + \mathcal{V}(\mathcal{A}_{(1:m-1\setminus m)\cup\mathcal{C}})\\ \mathcal{V}(\mathcal{A}^m_{m\cap\mathcal{C}}) + \mathcal{V}(\mathcal{A}_{(1:m-1\setminus m)\cup\mathcal{C}}).\end{cases}\\ &= \max\begin{cases}\mathcal{V}(\mathcal{A}^m_{(m\cap m-1)\setminus\mathcal{C}}) + \mathcal{V}(\mathcal{A}_{(1:m-1\setminus m)\cup\mathcal{C}})\\ \mathcal{V}(\mathcal{A}^m_{m\cap\mathcal{C}}) + \mathcal{V}(\mathcal{A}^{m-1}_{m-1\cap\mathcal{C}}) + \mathcal{V}(\mathcal{A}_{(1:m-2\setminus m-1)\cup\mathcal{C}})\\ \mathcal{V}(\mathcal{A}^m_{m\cap\mathcal{C}}) + \mathcal{V}(\mathcal{A}_{1:m-3}, \mathcal{A}_{m-2}\setminus\mathcal{A}_{m-1})\end{cases}\\ &= \max\begin{cases}\mathcal{V}(\mathcal{A}^m_{(m\cap m-1)\setminus\mathcal{C}}) + \mathcal{V}(\mathcal{A}_{(1:m-1\setminus m)\cup\mathcal{C}})\\ \mathcal{V}(\mathcal{A}^+_{m\cap m-1\cap\mathcal{C}}) + \mathcal{V}(\mathcal{A}_{(1:m-2\setminus m-1)\cup\mathcal{C}})\\ \mathcal{V}(\mathcal{A}^m_{m\cap\mathcal{C}}) + \mathcal{V}(\mathcal{A}_{1:m-3}, \mathcal{A}_{m-2}\setminus\mathcal{A}_{m-1})\end{cases}\end{aligned}$$

as required. □

Proof of Theorem 5.8. We prove the case $|\mathcal{C}| = 1$; the case $|\mathcal{C}| > 1$ extends easily from this. The idea is to apply the recursive formula again to the 'big' terms in (18), and show that they simplify to three terms, and that their structure is similar to the initial three terms. This will then imply that at each stage we need only keep track of three terms, and hence the recursion will be efficient. We have

$$\begin{aligned}\mathcal{V}(\mathcal{A}_{1:m}) &= \max\begin{cases}\mathcal{V}(\mathcal{A}^m_{m\setminus m-1}) + \mathcal{V}(\mathcal{A}_{1:m-1})\\ \mathcal{V}(\mathcal{A}^m_{m\cap m-1}) + \mathcal{V}(\mathcal{A}_{(1:m-1\setminus m)\cup\mathcal{C}})\\ \mathcal{V}(\mathcal{A}^+_{(m\cap m-1)\setminus\mathcal{C}}) + \mathcal{V}(\mathcal{A}_{1:m-2})\end{cases}\\ &= \max\begin{cases}\mathcal{V}(\mathcal{A}^m_{m\setminus m-1}) + \max\begin{cases}\mathcal{V}(\mathcal{A}^{m-1}_{m-1\setminus m-2}) + \mathcal{V}(\mathcal{A}_{1:m-2})\\ \mathcal{V}(\mathcal{A}^{m-1}_{m-1\cap m-2}) + \mathcal{V}(\mathcal{A}_{(1:m-2\setminus m-1)\cup\mathcal{C}})\\ \mathcal{V}(\mathcal{A}^+_{(m-1\cap m-2)\setminus\mathcal{C}}) + \mathcal{V}(\mathcal{A}_{1:m-3})\end{cases}\\ \mathcal{V}(\mathcal{A}^m_{m\cap m-1}) + \max\begin{cases}\mathcal{V}(\mathcal{A}^{m-1}_{m-1\setminus(m-2\cup m)}) + \mathcal{V}(\mathcal{A}_{1:m-2})\\ \mathcal{V}(\mathcal{A}^{m-1}_{m-1\cap m-2}) + \mathcal{V}(\mathcal{A}_{(1:m-2\setminus m-1)\cup\mathcal{C}})\\ \mathcal{V}(\mathcal{A}^+_{(m-1\cap m-2)\setminus\mathcal{C}}) + \mathcal{V}(\mathcal{A}_{1:m-3})\end{cases}\\ \mathcal{V}(\mathcal{A}^+_{(m\cap m-1)\setminus\mathcal{C}}) + \mathcal{V}(\mathcal{A}_{1:m-2})\end{cases}\\ &= \max\begin{cases}\mathcal{V}(\mathcal{A}_{1:m-2}) + \max\begin{cases}\mathcal{V}(\mathcal{A}^m_{m\setminus m-1}) + \mathcal{V}(\mathcal{A}^{m-1}_{m-1\setminus m-2})\\ \mathcal{V}(\mathcal{A}^m_{m\cap m-1}) + \mathcal{V}(\mathcal{A}^{m-1}_{m-1\setminus(m-2\cup m)})\\ \mathcal{V}(\mathcal{A}^+_{(m\cap m-1)\setminus\mathcal{C}})\end{cases}\\ \mathcal{V}(\mathcal{A}_{(1:m-2\setminus m-1)\cup\mathcal{C}}) + \max\begin{cases}\mathcal{V}(\mathcal{A}^m_{m\setminus m-1}) + \mathcal{V}(\mathcal{A}^{m-1}_{m-1\cap m-2})\\ \mathcal{V}(\mathcal{A}^m_{m\cap m-1}) + \mathcal{V}(\mathcal{A}^{m-1}_{m-1\cap m-2})\end{cases}\\ \mathcal{V}(\mathcal{A}_{1:m-3}) + \max\begin{cases}\mathcal{V}(\mathcal{A}^m_{m\setminus m-1}) + \mathcal{V}(\mathcal{A}^+_{(m-1\cap m-2)\setminus\mathcal{C}})\\ \mathcal{V}(\mathcal{A}^m_{m\cap m-1}) + \mathcal{V}(\mathcal{A}^+_{(m-1\cap m-2)\setminus\mathcal{C}}).\end{cases}\end{cases}\end{aligned}$$

This is exactly the desired form because the 'big' terms in the final equation are of the same form as those in (18), but with the indices decreased by 1. Therefore, the recursion is efficient. The running time is $O(N)$ since we scan each item at most three times to compute $\mathcal{V}(\mathcal{A}_{1:m})$. □